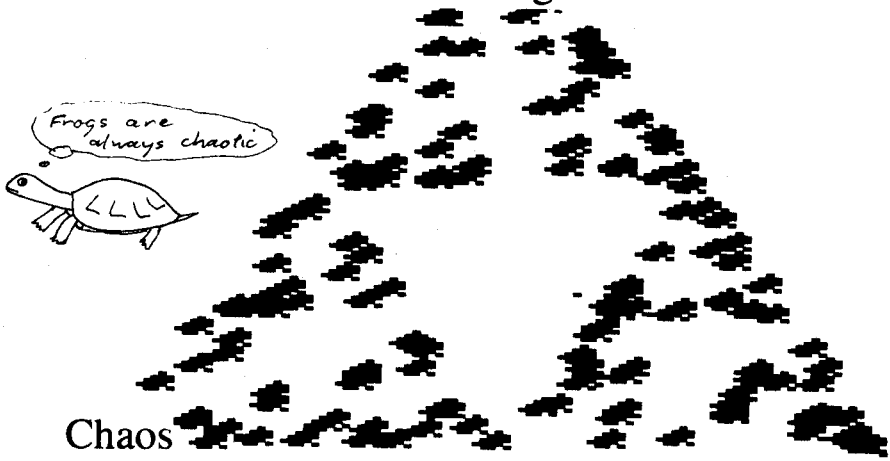
POALL

A Journal for Logo Users



Chaos

Volume 5 Number 2, May 1990

At last, made it. My thanks to Carolyn Dowling and Erroll Chopping for permission to reprint material from their publications. Without them, this would be a very thin edition.

WCCE '90 approaches fast. Some big names in Logo will be there: Seymour Papert, Steve Ocko, Mitchel Resnick, Andy di Sessa, Brian Silverman, Dan and Molly Watt, Barry Newell, Bruce McMillan, the list goes on..., and all speaking either at the conference itself or the pre-conference workshops. Alan Kay, originator of SmallTalk and GUIs too. Will be quite an occasion.

Peter

Peter J. Carter, Editor

POTS

Address subscriptions and items for publication to: The Editor, *POALL*
Plympton High School
Errington Street
Plympton S. Aus. 5038

Tur-mites

Tur-mites, according to A. K. Dewdney, in the September 1989 issue of Scientific American, are 'squarish, cybernetic creatures that have the most rudimentary of brains.' Even so, the patterns they create as they move about rival the architecture of real insects.

Tur-mites were inspired by the work of Greg Turk, who was experimenting with two-dimensional Turing machines. A normal Turing machine works with a one-dimensional tape as its 'memory', a tur-mite uses a plane divided into squares. Turing machines operate on a cycle:

1 Read the symbol under the read/write device.
2 Look up the table entry given by the current state of the machine and the symbol just read.
3 Write the new symbol given by the table, move the tape in the direction given, and enter the new state shown.

Tur-mites do much the same thing, except that they turn right or left and move across their flat universes, and colour squares instead of writing 0s or 1s. The patterns they make as they move appear random, chaotic, but the creatures themselves are rigidly deterministic.

The simplest tur-mite, which has only one internal state, follows this table:

Black	Red
Red, Left	Black, Right

In various Logo versions, it becomes:

LogoWriter[1]:

```
to turmitel
ifelse colorunder = 3 [setc 0 rt 90][setc 3 lt 90]
square
fd 3.8
turmitel
end
```

Apple Logo //:

```
TO Turmitel
SETPC 3
IF DOTP POS [SETPC 0 RT 90][SETPC 3 LT 90]
Square
FD 3.8
Turmitel
END
```

Both versions:

```
TO Square
PD RT 45 BK 2.123
LT 45 FD 3
LT 45 BK 4.246
RT 45 FD 3
RT 45 BK 2.123 PU
END
```

[1] LogoWriter procedures are conventionally written in all lower case, as distinct from the style normally used in *POALL*.

Phantom Fish Tank:

```
TO Turmite1 :dirn :x :y
MAKE "colour GETSTATE :x :y
IF :colour = 3 [SETSTATE 0 MAKE "dirn :dirn + 1]
     [SETSTATE 3 MAKE "dirn :dirn - 1]
Move
Turmite1 :dirn :x :y
END

TO Move
CELL :x :y
IF :dirn < 1 [MAKE "dirn 4]
IF :dirn > 4 [MAKE "dirn 1]
IF :dirn = 1 [MAKE "y :y - 1]
IF :dirn = 2 [MAKE "x :x + 1]
IF :dirn = 3 [MAKE "y :y + 1]
IF :dirn = 4 [MAKE "x :x - 1]
IF :x > 39 [MAKE "x 0]
IF :y > 39 [MAKE "y 0]
IF :x < 0 [MAKE "x 39]
IF :y < 0 [MAKE "y 39]
END
```

The next example is more complex, with two states, A and B, and follows this table:

	Black	Green
A	Green, Left, A	Black, Forward, B
B	Green, Right, A	Green, Right, A

This tur-mite produces a spiral pattern, and to make it work with AL //, a `ColourUnder` procedure can be used. It works well enough for our purposes, although it's colourblind[2]:

```
TO ColourUnder
OP ColourUnderAux 5
END

TO ColourUnderAux :colour
SETPC :colour
IF DOTP POS [OP :colour]
OP ColourUnderAux :colour - 1
END
```

```
TO Turmite2 :state
MAKE "colour ColourUnder
IF AND :colour = 0 :state = 1 [SETPC 5 LT 90 MAKE "newState 1]
IF AND :colour = 5 :state = 1 [SETPC 0 MAKE "newState 2]
IF AND :colour = 0 :state = 2 [SETPC 5 RT 90 MAKE "newState 1]
IF AND :colour = 5 :state = 2 [SETPC 5 RT 90 MAKE "newState 1]
Square
FD 3.8
Turmite2 :newState
```

For the LogoWriter version, spell `colorunder` correctly and change `SETPC` to `setc`.

[2] To explain why would take too much space here. Ask someone who understands the weird workings of the Apple's memory and graphics to tell you about it sometime.

The Phantom Fish Tank version is the best of the lot, although you won't be able to format it like this:

```
TO Turmite2 :state :dirn :x :y
MAKE "colour GETSTATE :x :y
IF AND :colour = 12 :state = 1 [SETSTATE 0 MAKE "newState 2]
IF AND :colour = 0 :state = 1
      [SETSTATE 12 MAKE "newState 1 MAKE "dirn :dirn - 1]
IF AND :colour = 0 :state = 2
      [SETSTATE 12 MAKE "newState 1]MAKE "dirn :dirn + 1]
IF AND :colour = 12 :state = 2
      [SETSTATE 12 MAKE "newState 1 MAKE "dirn :dirn + 1]
Move
Turmite2 :newState :dirn :x :y
END
```

Yet a third species has been discovered, with only one state, and following these rules:

Black	Red	Yellow	Green
Red, Right	Yellow, Right	Green, Left	Black, Left

```
TO Turmite3
MAKE "colour ColourUnder
IF :colour = 0 [SETPC 5 RT 90]
IF :colour = 5 [SETPC 1 RT 90]
IF :colour = 1 [SETPC 4 LT 90]
IF :colour = 4 [SETPC 0 LT 90]
Square
FD 3.8
END

TO Turmite3 :dirn :x :y
MAKE "colour GETSTATE :x :y
IF :colour = 0 [SETSTATE 1 MAKE "dirn :dirn + 1]
IF :colour = 1 [SETSTATE 13 MAKE "dirn :dirn + 1]
IF :colour = 13 [SETSTATE 12 MAKE "dirn :dirn - 1]
IF :colour = 12 [SETSTATE 0 MAKE "dirn :dirn - 1]
Move
Turmite3 :dirn :x :y
END
```

Why three procedures rather than a generalised one with rules in the form of a list? Stack overflows. When elegance fails, try brute force.

Alan Turing devised the Turing machine, not as a real device, but as a theoretical one. If a computation could be reduced to a set of rules, he argued, then it could be done by a machine. The Turing machine is the logical foundation for all computers, and a tur-mite's 'brain' is, at least in theory, as powerful as any computer.

More obvious on the screen is the fact that a simple set of rules can give rise to the complex and unexpected.

References:

Carter, P. J. 'Turing Machines' in *POALL* Vol 2 No 2, April 1987
Dewdney, A. K. 'Computer Recreations' in *Scientific American* Vol 261 N0 3, September 1989
Hodges, A. *Alan Turing: The Enigma of Intelligence* Burnett Books 1983

Resources

LogoWriter Secondary

Late last year the Version 2 of LogoWriter became available in Australia, together with teaching materials for junior secondary level.

The original version of LogoWriter has been around for some four years, although it has not had the impact in South Australia that it deserves. As the name suggests, LogoWriter is a combination of Logo with a word processor. Some features of Logo, like property lists and the ability for procedures to redefine themselves were deleted, and the integrated word processor, while it has cut and paste, and search and replace facilities, is rather limited. Given the intended use however, the limitations are of little import, especially given the completely changed editing and filing.

A LogoWriter 'file' (the word is not used) is a 'page', and the 'catalog' is the 'contents page'. To load a page, one selects it from the contents page with the arrow keys and presses <Return>. A page has two sides, the front, on which the four Turtles go about their graphics and word processing is done, and the 'flip side' , the editor, accessed simply by ⌘ F. Saving, after the page is named with `namepage`, is by pressing <Escape>. There is none of the `ED "wotsit`, `LOAD "thingie` or `SAVE "thisthing` which caused so much frustration and anguish to younger users in the past.

LogoWriter documentation is excellent, as would be expected of the top Logo developers. The idea of teaching materials for Logo might sound, at first, a contradiction, but the booklets for primary and secondary levels find the elusive balance of direction and free discovery. The suggested projects in the five books (plus Teacher's Guide) range from animation, through adventure games, to music. My Year 9s are currently finding them both interesting and easy to follow, and are pleased with the results they are achieving.

The other point about LogoWriter documentation is that the same manuals are used for the Apple, MS-DOS and Commodore 64 (1.1) versions, because LogoWriter was originally designed to be as similar as possible on all three systems. The latest version is beginning to diverge somewhat, on the Apple for instance, now using ProDOS instead of the earlier unique DOS. There is a full set of ProDOS commands, but you do have to remember to use % instead of / in pathnames. Version 2 also requires 128 k, and as a bonus gives access to memory with `.examine` and `.deposit`. The Apple Logo // routines for driving the Tasman Turtle were loaded straight into LogoWriter, edited (three input `IF` is now `ifelse`, and free variables are not saved), and ran first time.

MS-DOS users also have access to DOS. There are utilities on the disk to make working disks tailored to the system in use, CGA, EGA, or PS/2 graphics, and a range of printers. PS/2 machines can allegedly display 256 colours, although 8 of them are black.

The word from Logo Computer Systems Inc is that there will be another new version for the Apple IIGS, to make full use of the machine's memory, graphics and sound. It will also incorporate LEGO TC Logo primitives. (The current TC Logo is a 'cut down' version of LogoWriter.)

Any primary school still using a conventional Logo should really consider changing to LogoWriter, and the same is true for junior secondary classes. LogoWriter has all the power necessary, and is so much easier for children to use. If your local dealer can't supply, call the new LCSI agents in Australia, Computelec Data Systems, (03) 786 7177. Prices: Site/Network license (unlimited copies) $695, Lab Pack (6 disks) $450, Single disk with materials $250, Materials kit (no disk) $180.

Tom Forester: *Computers in the Human Context*

Readers of the Tuesday edition of The *Australian* will be familiar with the work of Tom Forester, Lecturer in the School of Computing and Information at Griffith University in Queensland. Forester's articles effectively dispel many of the myths of the 'computer revolution', and pour scorn on much of the 'hype' surrounding much of it.

His latest book contains more than 40 essays by a wide variety of authors, all experts in their fields, and is in four parts: Computers and Society, Computers and People, Computers and Organisations, and Computers and the Future.

Computers and People includes the section IT and Schools, with articles by Alison Bass, and Donald Ely (University of Syracuse, NY) and Tjeerd Plomp (Twente University, Netherlands).

Ely and Plomp view the use of computers in the context of 50 years of history of educational technology, a history largely of failure. They believe the difficulties of the past were caused by confused goals on the part of participants, and an emphasis on the medium, rather than on learning. There was also resistance to change, lack of support systems, and lack of skills on the part of teachers. Add to that the expense of the equipment/training/support/etc. and the lack of quality software. It all sounds familiar. Ely and Plomp conclude that there is little evidence of a technological revolution in education.

Alison Bass analyses 'Project Headlight' at the Hennigan school in Boston. With quotes from Papert, she gives an outline of the philosophy and history of LOGO (*sic*), as well as some personal history of Papert himself. Seymour Papert's early years were spent in Swaziland, where his father was an entomologist, researching the tsetse fly. It was in the camps that the young Seymour became fascinated by the gears of the trucks and other machinery.

Papert was also surrounded by black children, and it was a shock to the ten year old when the family moved back to Johannesburg, and segregation. An attempt to set up night classes for black servants was met with firm refusal. The illogicality set him thinking about the nature of thought.

It was some 20 years later that Papert met Piaget:

> "Piaget brought many things together for me," Papert says. "Before I met him, I had been intellectually torn between my interest in how people came to think and my interest in more abstract ideas. Piaget showed me a way in which my caring for math, for the philosphy of thinking, and for social reform all seemed to go together."'

There are quotations also from critics Joseph Weizenbaum, and Hubert and Stuart Dreyfus (There is also an essay by the Drefuses in the book: 'Why Computers May Never Think Like People'), who argue that the logical step-wise analysis form of thinking learned by programming is only one (limited) form of human thinking.

Bass concludes with with an enthusiastic quote from one of the Hennigan teachers: 'Project Headlight has done some wonderful things for our kids. If it were up to me, it would never end.'

Forester's book is too heavy for school students, but it would be essential reading for anyone concerned with the effects of computers on society. It presents a realistic view of, to use its subtitle, Information Technology, Productivity, and People. The publisher is MIT Press.

James Gleick: *Chaos*

Chaos is not the state of your classroom, but a very respectable theory that may explain all manner of disparate things. Gleick's book is written for the non-technical, non-mathematical reader, and begins with an account of Edward Lorenz's meteorological simulations, computer models of the atmosphere that turned out to be quite unpredictable. Lorenz was to coin the term 'butterfly effect' to describe how minuscule vortices could ultimately affect the global circulation.

He moves on to the work of Sydney born biologist, Robert May, who was investigating the stability of populations. By merely slightly changing one value in the equation, the results began to oscillate wildly, a steady state suddenly bifurcated, and then became chaotic.

May's equation, $x_{next} = rx(1-x)$, translates into Logo as:

```
TO May
CLEARSCREEN HIDETURTLE WINDOW
Eq 2.5 0.5
END
```

```
TO Eq :rate :x
MAKE "x :rate * :x * ( 1 - :x )
IF ( ABS ( 100 * :x ) - 50 ) < 135
      [DOT SE ( 100 * :rate ) - 400 ( 100 * :x ) - 50]
Eq :rate + 0.001 :x
END
```

This will show about as much as May first saw on a computer, before it crashes with either a SETPOS DOESN'T LIKE ..., or NUMBER TOO LARGE error. Like May, you'll need more computing power to see the real chaos that ensues.

Gleick moves on, with Mandelbrot and fractals, and pages of coloured images of the Mandelbrot set, Ruelle and turbulence, Mitchell Feigenbaum and a universal theory, Lichaber and liquid helium, and so on.

What began as isolated, puzzling problems for resarchers in many fields, eventually coalesced into the modern understanding of chaos: apparently simple things contain within them the seeds of chaos, what appears to be predictable, may be totally unpredictable.

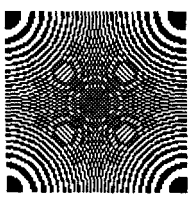
Martin Gardner, who himself has done as much as anyone to promote an understanding of fractals and other ideas in mathematics and science, describes Gleick's book as:

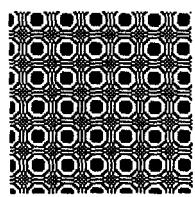
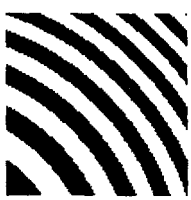
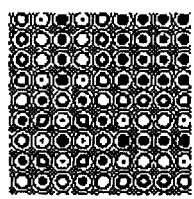
> 'The first popularly written book about this fascinating, rapidly growing discipline. It is a splendid introduction. Not only does it explain accurately and skilfully the fundamentals of chaos theory, but it sketches the theory's colourful history, with entertaining anecdotes about its pioneers and provocative asides about the philosophy of science and mathematics.'

Well worth reading, and readily available at most good bookstores. It's in paperback, published by Cardinal. ABC-TV has twice shown a programme covering much of the material, and of course there was the conference in Sydney in February, reported in The *Australian* if nowhere else. The 1990 edition of the Encyclopædia Britannica *Yearbook of Science and the Future* contains a chapter entitled 'Chaos: Does God Play Dice?', eminently readable and well illustrated.

Another book on a similar subject is *The Cosmic Blueprint*, by Paul Davies, newly migrated to Australia, to Adelaide University in fact. Davies has established a reputation as an author, and this book should also be an interesting and readable one.

Just received is *The Turing Omnibus*, by A. K. Dewdney. Sixty one short chapters on a variety of topics, aimed at the tertiary level. More on it next time, but in the meantime, some images from the Chapter 1 algorithm:



Chaotic ravings from Errol Chopping

Frogs in the Real World

A few years ago, while attending a State Computer Education Conference, I asked a representative of a software company if a version of the Logo programming language was available for my type of computer.

The reply was interesting. I can still remember the actual words he used: "We haven't bothered with Logo - it has no relevance to The Real World."

Since that conversation, the expression "The Real World" has caused me a little disquiet. The expression appears in news reports, in assignment essays I have read, and in television documentaries. It seems to have become a favourite throw-away line for many people and the more I hear it, the more I get the impression that its use is meant as put-down for any activity in the Education field.

There are times when I want to point out that all facets of Education have relevance to this utopian "Real World". Schools are real things, Pupils and Teachers are real too. The contribution that computers are making in Education is indeed real and highly relevant to the success of our pupils in whatever field they wish to persue.

A good comparison can be made between computing in the "Real World" and the computing that takes place in schools. In fact, if many of the current practices in "Real World" computer use were to form the basis for a Computing Studies Syllabus, then we would finish up with a far less rigorous, less comprehensive and less relevant course than is currently operating. The interpretation of "Real World" computing I'm concerned about here is the one that suggests our school courses should focus strongly on the purely vocational aspects of computing. The truth is that the majority of computer users in the "Real World" have only a surface knowledge of computing. They are trained only to recall the location of the On-Off switch from whence they perform a series of unsophisticated mechanical steps involving a single piece of software. This they often do all day, every day.

In contrast, School courses aim at providing skills and understanding on a broad range of software applications, operations and solutions; and these to a depth which is well beyond the mechanical. In short, we are attempting to Educate our pupils rather than to Train them.

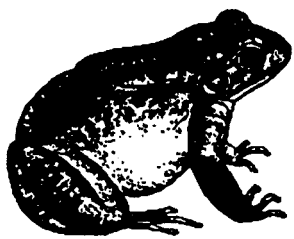
To further emphasise some of the differences between vocational and educational computing, consider another quote by a computing professional I have heard: "Teachers, by definition, have inferior computing equipment." While I can't argue conclusively with this statement, (school resources are another issue) I disagree with the implication that what is being done in schools might therefore be inferior and the phrase "by definition" also causes me some concern. In fact, with the resources teachers have, their training and expertise, and with both the junior and senior computing studies courses to guide them, schools are starting to provide a higher level of computer literacy than is observable in most "Real World" settings.

In order to (hopefully) tie these threads together, I'll finish by bringing together an

Reprinted from *C.E.G. Newsletter* (NSW), December 1989.

Chaotic ravings from Errol Chopping

example of School computing, Logo programming, and up-to-date research which is certainly within the reach of our pupils. It involves the puzzle I left for readers at the end of my last article in this August publication - the problem of the Jumping Frog. I am aware that by now, many of you will have seen the television documentary on "Chaos" which has already provided a solution. Nevertheless, here is the problem again, with the computer solution as promised.



The Problem:
A Frog starts from any random location. It chooses, at random, one of the three vertices of an equilateral triangle and jumps half way to it leaving a mark on the ground where it lands. Starting at this position, the Frog again chooses at random a vertex of the triangle, jumps half way to it and again leaves a mark on the ground where it lands. The Frog's seemingly chaotic behaviour continues forever. What pattern is eventually formed from all the markings on the ground?

This solution is written in the very same Logo implementation that was considered as not relevant to the "Real World" a couple of years ago and will provide an image on your computer screen which allows you to see the Frog's chaotic pattern. The code will be compatible with all versions of Logo with minimal changes necessary.

A Solution:

```
TO frog
cs ht
triangle 200
getready
jump
END

TO triangle :size
make "V1 [-90 -60] dot :V1
pu setpos :V1 setheading 30
fd :size make "V2 pos dot :V2
rt 120 fd :size make "V3 pos dot :V3
END

TO getready
make "x list ( random 260 ) - 130
 (random 180 ) - 70
setpos :x dot :x
END

TO jump
make "x midpoint :x thing word "V
 whichvertex
dot :x
jump
END


TO midpoint :x :y
op list ( sum first :x first :y ) / 2
 (sum last :x last :y ) / 2
END

TO whichvertex
op 1 + random 3
END
```

Best wishes
Errol. ☆

Brian Silverman talks with COM 3 . . . about Logowriter

COM 3: A lot of schools, primary schools in particular, have had Logo now for maybe six or seven years. Some of them have recently switched over to Logowriter or are thinking about doing so, and others don't really know because they don't know what the difference is.

Silverman: Logowriter's like Logo, only better.

COM 3: Better in what way?

Silverman: The real answer to that is that it's basically better because it's newer. What happened was that our company made Apple Logo in about 1981/82, and that was based on a lot of research that was done at MIT. We let Apple Logo out into the world for about five years, and at the end of five years it was critically evaluated — what worked in Logo, what didn't. So what we did with Logowriter is emphasised the things that were really working; we made it easier to use.

COM 3: Which did you see as the things that were really working?

Silverman: Certainly the graphics, though the graphics in Logowriter are better. The other thing that we did is there were a number of things we tried and tried to explain in Logo but we couldn't, so in Logowriter we redesigned it to make the questions go away. There were certain problems we couldn't solve in Logo, so we made them dissolve in Logowriter.

COM 3: Can you give any examples?

Silverman: Yeah. Most kids and teachers for example have a hard time understanding work-space . . the notion of what was in the workspace and what was in a file was always very confusing. You would say save "SQUARE and the computer would say 3 PROCEDURES DEFINED. It included your triangle, and no-one really understood that.

We tried logically explaining it and kind of threw our arms up in the air and gave up, and figured that maybe it's better to just dissolve the distinction between the workspace and a file. So in Logowriter we tried out the scrapbook idea and the pages with flip sides, and we find that that works a whole lot better. For people who've used Logo before . . . kids who've used Logo before take to Logowriter and understand it after about a day or two; teachers understand it after about a week or two. And for new users, they get into Logowriter much more easily than traditional Logo.

COM 3: For people who are wondering whether to change over or not, what would be the areas that could present some difficulties?

Silverman: I think it's mainly that we made a few things simpler that people had spent a lot of time getting to understand. Then when the concept suddenly disappears, it's a bit jarring. It's unfamiliar; it's a slightly different environment. Anything that's a bit different takes a certain time, it's confusing, but I don't think that there's anything substantial that's different. The changes are to make things easier.

COM 3: The addition of the word processing, that's probably the most obvious change to people who have just come into Logowriter.

Silverman: Right. What Logo was about to begin with was the notion of doing projects, and the sorts of projects that were done with the earlier Logos tended to be graphics and occasionally maths, but primarily graphics. When we add the word processor, suddenly the projects become graphics and text, and the page system allows there to be animation as well. What we found was that the people who were attracted to Logo at the beginning tended to be maths and computer science teachers, and with Logowriter it's as often as not teachers in social studies and English and history classes and places like that.

So Logo appealed to people who were already more or less convinced that the computer was a good thing. With Logowriter there's more of a tendency for people who otherwise would have ignored the computer to look at it and see something that may mean something to them.

COM 3: What age or grade ranges would use Logowriter in North America and Canada?

Silverman: The software isn't particularly targeted at any age range. We've done three different sets of material to go with it. The original set of materials was aimed at about grades 4 to 7 and was extended upward and downward a little bit, then about a year and a half later we came out with what we called the Primary materials, which were for grades 1 to 3. Above that we're coming out with our Secondary materials, which are for grades 8 and 9, perhaps 10. It's been very successful in most grades below 9 up until now, and we're hoping that our Secondary materials will push up the age range a bit.

One of the things that I guess I could say is that Logowriter is currently being used in a large number of North American schools. Somewhere between 10 and 20 per cent of all schools in North America have it and are using it very actively.

COM 3: Is there anything that more advanced students, say year 11 and 12 computer science students, could do in plain Logo that they're not able to do in Logowriter? Have any things that are significant at that level been cut out?

Silverman: There have been things cut out, and there's some question as to whether or not they're significant! Year 11 and 12 computer science-type students may be missing some things. Another thing that we realised in the early days when we were doing the original Logo is that there are really two different views of Logo. There's Logo the programming language and there's Logo the educational environment. Well, in the early days we thought that we could just design something that would be both, and by a few years later we realised that this led to some contradictions, so we decided that with Logowriter we'd be building Logo the educational environment, even if it would be a bit at the cost of Logo the programming language. As it turns out, Logowriter is easy to program and easier to use than the other Logos, but there are some things missing, most notably if you're doing data file projects. Logo II and things like that have a good data file manipulation system, but Logowriter doesn't have it all.

COM 3: Going back some years, I seem to remember that there was a fairly scathing reaction from some of the original Logo people to the proliferation of Logo support materials that were produced. I mean, Logo had been envisaged as a free environment for people to create and explore their own projects, and suddenly here were all these printed worksheets and sets of instructions coming out. Now it's turned around, and the worksheets and other materials are coming out as part of the Logowriter package. Was that a very big change of attitude?

Silverman: Not really. I think that the scathing reaction was to the kinds of written materials that I think we tried to avoid in Logowriter. There have been dozens and dozens of books in the last eight years. I guess since **Mindstorms** and since Apple Logo came out there have probably been between a hundred and a hundred and fifty books written about Logo. Of those there are ten or twelve good ones. There are a lot of them about, you know, how to make a square, and it's not obvious that that helps a whole lot. What we tried to do with the Logowriter materials was tread a fine line between making the teachers feel comfortable that they're not being left alone with this weird software, and leaving it open-ended for kids. Things like, in Logowriter materials we've avoided taking photographs of the screen. Pretty much all of the pictures are fuzzy line drawings done by our graphic artists, because that way teachers couldn't hold it up and say "This is the right answer". It just can't look the same as the one in the book.

COM 3: So each work card is really just trying to point the kids in the direction of a new tool that they can then use for doing whatever they want.

Silverman: Yes, mostly we were hoping that the materials would be taken as starting points rather than as an exact set of steps to go through.

COM 3: Is plain Logo still being produced and is it being modified in any way? Is it still an active, on-going thing, or has Logowriter really replaced it?

Silverman: As far as LCSI is concerned, Logowriter has really replaced it. We're still supporting Apple Logo II, but we're not likely to make any great changes to it in the future. I think that the direction of our company is very much that we've decided that we want to be an educational materials company more than a computer software company.

COM 3: And has the reaction been that Logowriter's much easier to use in schools and more appropriate than the original Logo?

Silverman: Very much so. Logo became faddish in about 1982, 1983 in North America, and at that point everybody just had to have it. By a couple of years later, by 1985, a lot of people had decided that they'd "done" Logo and it was time to move on. Logowriter was like the rebirth of the Logo movement. There's a different group of teachers that are using it. Teachers that considered themselves to be much less at the technological leading edge, but who believed in the ideas in **Mindstorms**, and people who just thought that the computer could be used to do something reasonable.

Let me tell you a bit about the Costa Rica project. Costa Rica decided to put computers in all of the schools in the country, and they asked all of the major computer manufacturers to say what they would do, given the chance to put computers into all the schools. IBM asked Seymour Papert to write their proposal for them, and it was finally accepted, so there's essentially a pilot project been going on in most of the schools in Costa Rica which is pretty much based around Logowriter. That's what they're doing with the computer in that country and it's actually the case in a number of places, a number of jurisdictions, that what they do with their computers in schools is Logowriter, LEGO/Logo and almost nothing else.

COM 3: Well, if that's a rich enough environment I guess it's what ought to happen. Some people here used to say that you only needed Logo, then they said you only need Logo and a word processing program, which I guess is what you've given them, then it was Logo, word processing and a database. . .

Silverman: Right, but I'd argue with that because the sorts of activities that could be done with the database I would claim were better done with something like Logowriter.

COM 3: Could you explain what you mean?

Silverman: Sure. Logowriter isn't a database, and you'd have to ask the question, what are you using a database for? It's to collect and organise information you've pulled

out from various places. If you use a database, it's one more tool that you have to learn about to be able to use, whereas on the other hand if you want to keep your data in just a word processor, it's much simpler. The inconvenience of keeping it in a traditional word processor is that there's not much you can do with it, but because Logowriter's a word processor with a built-in programming language, you know you can put your data into the word processor and it's still active enough that you can do something useful with it.

COM 3: Do any of the support materials suggest how you might do that?

Silverman: Not in as much detail as they could. But why don't I give an example? There's a 5th grade science teacher in Saint Paul, Minnesota, who was working with her kids, and the data they were collecting were about all of these microscopic bugs. They got together and made a collection of several thousand Logowriter pages of pictures of various bugs, and wrote on each page some of the properties of each, then on one of the pages it had a kind of summary, like if it's got a little hairy thing sticking out of its sides then it must be one of these. And they basically collected a bunch of interesting data like that. A different group has in fact used LEGO/Logo for the same sort of thing, to collect weather data. They just put some instruments up on the roof of their school and fed it back into the computer, and they collected how fast the wind was going or what the temperature was over a period of a few weeks. Once they got all the data they used the programming part of Logo to analyse it and make graphs and do all the other good things with it.

COM 3: I can see how you'd go about collecting and ordering your data and putting it in, but what about retrieval? I suppose you use SEARCH, or something like that.

Silverman: Yes. It's simple. To finish up by getting back to your earlier question about whether we're supporting Logo . . . Logowriter as it stands now isn't everything we wanted it to be, because we felt it was very important to be able to run it on an Apple IIe with 64k of memory. In the future we'll be coming out with more advanced versions of Logowriter. We'll put it on a computer with more memory that runs faster, and so have a Logowriter with more memory that runs faster. That's not to say that the Apple II version isn't good enough. We think that the Apple II version is great, though if you had more computer power then you could have more flexibility. We certainly intend to have Logowriter grow with the technology as it grows, while at the same time making sure that it does really good things on the technology that's the norm at any given point in time.

The Directed Turtle

All sorts of methods are used to introduce directed numbers to children. Here's a Logo program that draws the number line and then moves the Turtle along it as numbers are added and subtracted:

```
TO NLine :limit
CLEARSCREEN SPLITSCREEN HIDETURTLE WINDOW
MAKE "scale 120 / :limit
PU SETPOS [-120 -10] SETH 45 PD
V SETH 90
REPEAT (2 * :limit) - 1 [FD 120 / :limit Tick]
FD 120 / :limit SETH 225 V PU
SETPOS [-125 -20] GRTYPE WORD "- :limit
SETPOS [115 -20] GRTYPE WORD "+ :limit
SETPOS [0 -20] GRTYPE "0
HOME SETH 90 SHOWTURTLE
END

TO V
REPEAT 2 [FD 3 BK 3 RT 90]
END

TO Tick
LT 90 FD 2 BK 4 FD 2 RT 90
END
```

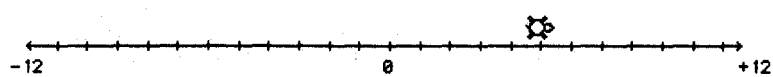
The 120 in `NLine` should be varied according to the screen width of your system, and `GRTYPE` is LCSI Logo II's primitive to put text on the graphics screen. The `WINDOW` is there to prevent spurious answers caused by wrapping around: take care with LogoWriter versions. Now for the mathematics:

```
TO StartAt :number
FD :number * :scale
PR SE "At XCOR / :scale
END

TO Add :number
FD :number * :scale
PR SE "answer: XCOR / :scale
END

TO Subtract :number
BK :number * :scale
PR SE "answer: XCOR / :scale
END
```

You could add enhancements like changing colour according to direction, or using a Turtle of a different shape. How about `Multiply` and `Divide` procedures?



'LOGO as a Prelude to Lisp: Some Surprising Results'

That's the title of a paper in the SIGCSE Bulletin of the ACM, Vol 21 No 3, Sept 1989, pp 35..38. The Author, Kenneth Louden, of the Department of Mathematics and Computer Science of San José State University, set out to test the suggestion that:

'a LOGO-based (*sic*) introductory course in computing would be preferable to a more traditional curriculum, based on the fact that (1) fundamental notions of computer science could be introduced quickly and naturally without the overhead of the complex syntactic and semantic rules of an imperative language, and (2) students would be better prepared to understand fundamental questions in the theory of computation, as well as in artificial intelligence.'

A one semester course was prepared and made a requirement for first year students, using the text by Burke and Genise, *Logo and Models of Computation: An Introduction to Computer Science*. After the course, students moved to the more traditional Pascal and Data Structures courses. Faculty members felt that the Logo course was of some benefit, and Louden decided to study, statistically, the results in the 'upper-division Lisp course' to see if there was an observable difference between Logo and Pascal trained students.

Student grades were analysed, as were the results of questionnaires, and Louden admits that the data collection is crude, not using a single instrument to rank students, and not controlling for extraneous factors. The final results are shown in Figure 1, which shows Grade Point Average for the sections of the LISP Course. Chi-Square and Anova tests revealed no significance.

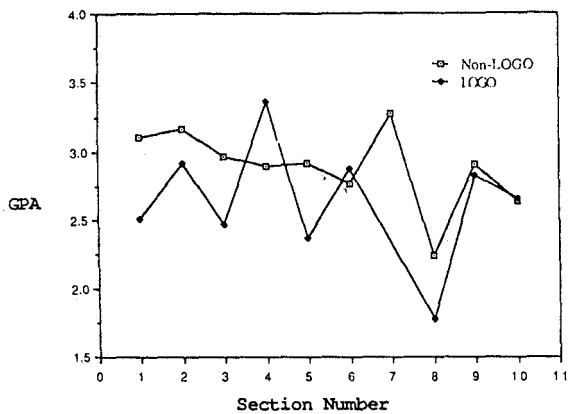


Figure 1: Grade Point Average by LISP Section

Louden concludes:

'The only effect that we saw in the above results ... was that those students who did not have an course in LOGO programming seemed to do a little better in a Lisp programming course than those who had taken LOGO programming, This is a strange result, and it is likely that this is due to some side-effect in the samples (possibly the age factor...).'

He conjectures that the differences may be due to the Logo group was faced with two languages in their first year (Logo and Pascal) instead of just one, or that first year students might not be ready for functional (as distinct from imperative) programming. It might also be that the non-Logo group performed better simply on the basis of maturity.

'Nevertheless, two things appear to be clear: the LOGO students certainly did not do any *better* than the non-LOGO students in the Lisp course, and, second, more study should be done before any claims should be made as to the effect of an introductory course using LOGO on students in a computer science curriculum.'

Interesting.

Galactic Cannibalism: Using supercomputers to model galactic dynamics.

Barry Newell, known to some as the Administrator of the Mt Stromlo and Siding Springs Observatories, and to others as the author of *Turtle Confusion* and *Turtles Speak Mathematics*, was guest speaker at the Australian Computer Society Branch Breakfast on April 27th.

Barry began by reminding his audience that it was the task of science to build models of the universe, or parts of it, and to test them as far as possible, so that they could be used for making predictions. (In essence, that is what education is also about, helping students build mental models, and to test and refine them.) The physical sciences, dealing with matter, could describe interactions, such as those in thermodynamics and aerodynamics, with equations, whose complexity increases as the number of particles being investigated increases. To deal with the increasing complexity, more computing power was needed: serial computers could deal with perhaps ten thousand particles, supercomputers were approaching a million particles in their calculations.

As it happens, calculations of the orbits of stars in colliding galaxies were easier than many problems in aerodynamics. Only one force was involved, gravity, and most interactions depended on the rate of collision relative to the rotation of the galaxies; the faster the collision, the less interaction.

Cosmological theories can now be tested, and the rule of testing was to see whether the simulations matched what we see in the sky.

Barry then showed a series of slides, showing solar turbulence and granulation, real and simulated, waves at a fluid boundary, and a shock wave in an interstellar medium. The final slides were telescopic views of colliding galaxies, the Mice, and one with 'shells'.

The next part of the multimedia presentation was a videotape, produced by the CalTech N^3 machine, a computer with 512 processors and cheaper than a Cray. Despite being faster than the Cray, the machine took some 120 hours for each of the simulations, which were animations of galactic collisions, some producing 'tails' like the Mice, others the faint shells. The latter were particularly interesting, as the remnants of the colliding mass of stars oscillated to and fro, throwing off expanding circles of matter.

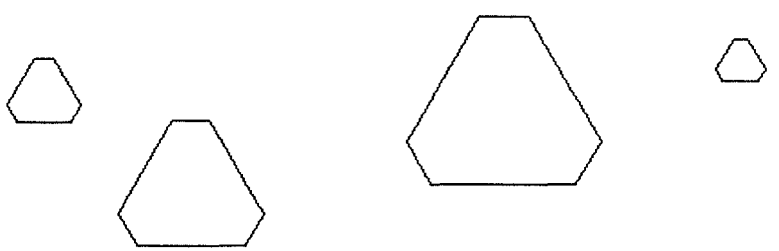
The final part was a film of the evolution of galaxies, condensing out of the filamentary structure of the universe; 10 to 15 billion years of history in a couple of minutes.

Barry's closing comment was that the telescope gave us a static view, a snapshot, of the dynamic universe, while computers could make a 'tremendous impact' on astronomy by allowing the dynamics to be explored.

Questions followed, the first being about the standing of Australian astronomy. The answer is that Australian astronomy is well supported and at a high standard. Barry had used the word 'chaos' on a couple of occasions, and he was asked what he meant. Barry's view is that the universe, however random and apparently unlawful (in the sense of following the laws of physics), is in reality unpredictable.

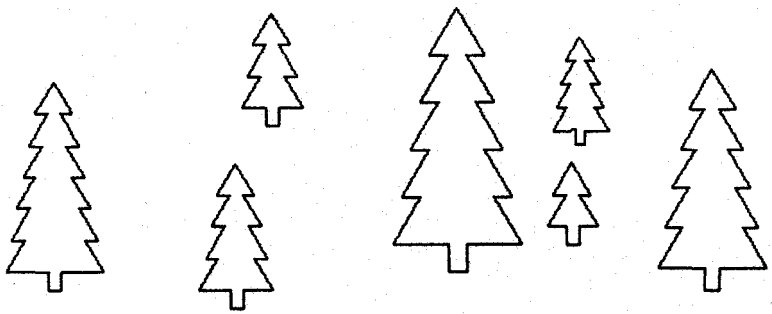
'Galactic Cannibalism'? In collisions between galaxies, the larger grows at the expense of the smaller, often completely absorbing it.

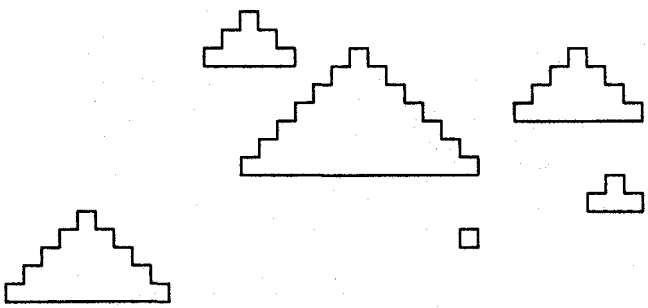
The food was good, too.

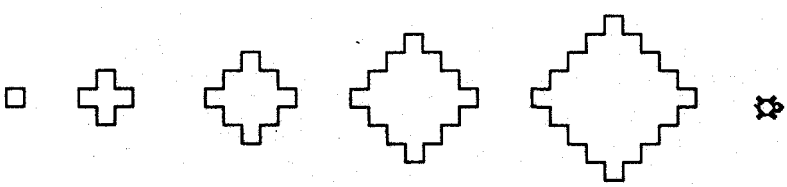


Puzzles

Here are some puzzles to keep minds and Turtles busy. In each case, the shapes are all drawn by the same procedure: only the inputs are different.





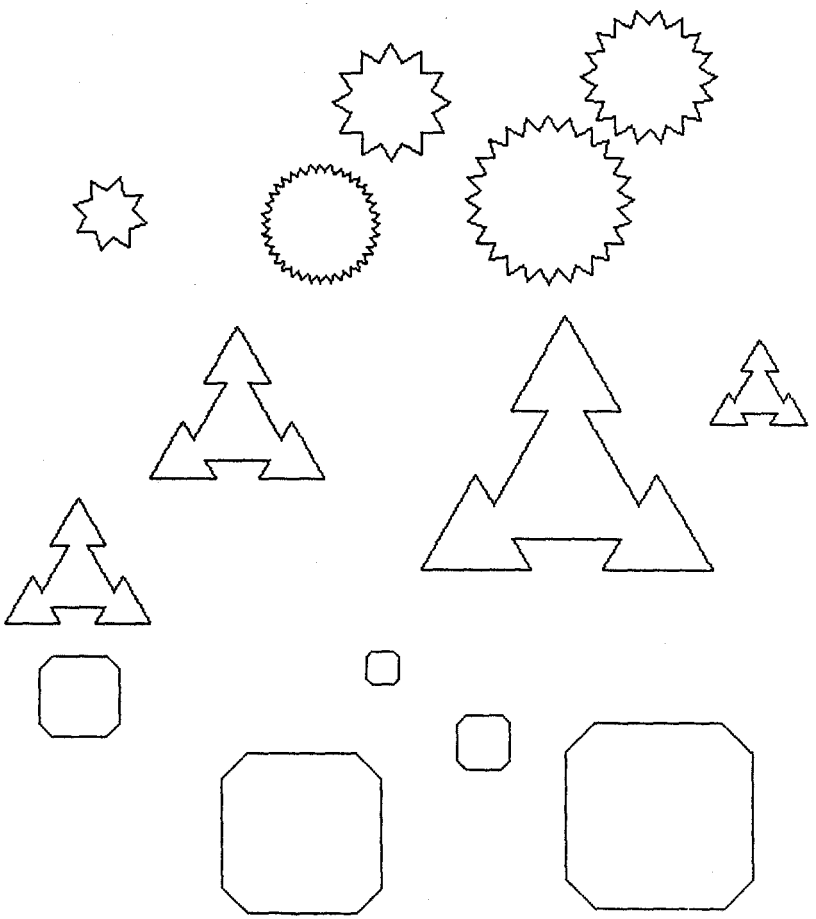


change for LogoWriter

LogoWriter's editing includes `search`, `replace`, `found?` etc., but not a global change. It's not difficult:

```
TO Change :from :to
TOP
ChangeAux :from :to 0
END

TO ChangeAux :from :to :count
REPLACE :from :to
IF NOT FOUND? [(SHOW "Finished: :count "changes) STOP]
ChangeAux :from :to :count + 1
END
```



Commuting At Entropy House

To send a process to the printer in a UNIX system the word is `lpr`. Here's an `lpr` for L/UX, the Logo toy UNIX (see *POALL* Vol 4 No 3). This one's for Apple Logo //%LCSI Logo II which you can adapt as required:

```
TO lpr :command
DRIBBLE 1
IF LISTP :command [RUN :command] [RUN SE :command []]
PR CHAR 12
NODRIBBLE
END
```

The VAX 121 isn't one of DEC's minicomputers, but a vacuum cleaner. In case you ever wondered, VAX means Virtual Address eXtension, and VMS, the operating system, which gets larger and slower with every release, means Virtual Memory System. Does the VAX 121 make the place virtually clean?

HyperLearning Forum is the newsletter of the recently formed HyperLearning Network, an organisation promoting the use of hypertext/hypermedia in education in general, the use of HyperStudio by Roger Wagner Productions in particular. HyperStudio is very like Apple's HyperCard, but it runs on the Apple IIGS, uses colour (640 mode graphics) and incorporates sound digitising hardware and software. There's no scripting (yet) but XCMDs are possible. PHS students are finding it easy to prepare stacks, and enjoy recording (and mutilating) sounds. Subscription is $US39, and includes a 'Best of HyperLearning' disk. Whether there will be any material on HyperCard, SuperCard, Guide, HyperPAD or other systems remains to be seen.

The address is HyperLearning Network, POB 103, Blawenburg, NJ 08504, USA. We can demonstrate HyperStudio for you at BiKiLog sometime.

Thinking Logo is now out of print: the last copies were sold to a school in Sydney the other week. Assorted doodles and scribbles exist for a second edition, but don't hold your breath. (I'm working on another new sea kayak design, and that takes lots of time, and room in the shed. It's a big, three dimensional cut and paste job.)

Brian Silverman (see elsewhere in this issue) and colleagues have had some good press in *Scientific American* in recent times. The October '89 issue has a description of the famous Tinkertoy computer built by Danny Hillis (of Connection Machines fame) and Silverman. It apparently played a mean game of tic-tac-toe. The 'Computer Recreations' column in the January '90 issue is devoted to cellular automata, using Phantom Fish Tank (Apples) and Rudy Rucker's CA Lab for MS-DOS PCs.

Nexus III is now up and running at Angle Park, and no, neither CEGSA, nor BiKiLog nor *POALL* is represented on it yet. Do you want to receive *POALL* and BiKiLog news on it? Write (on paper) with your requests/ideas/suggestions (including an article for the next issue).

POALL's improved look this edition is thanks to a HP DeskWriter printer. Same resolution as a laser, but it's all done by spitting drops of ink. It even prints the envelopes. *POALL*'s other printer, ie. Ken Anderson, who prints/collates/staples, is on sick leave, so other arrangements for this issue.

Talking envelopes, on the address label there's a number, like 0503. That number is the issue to which you're paid, and if it's your last issue, there'll be a red spot. Time to renew.