Scientific American February 1984

COMPUTER RECREATIONS

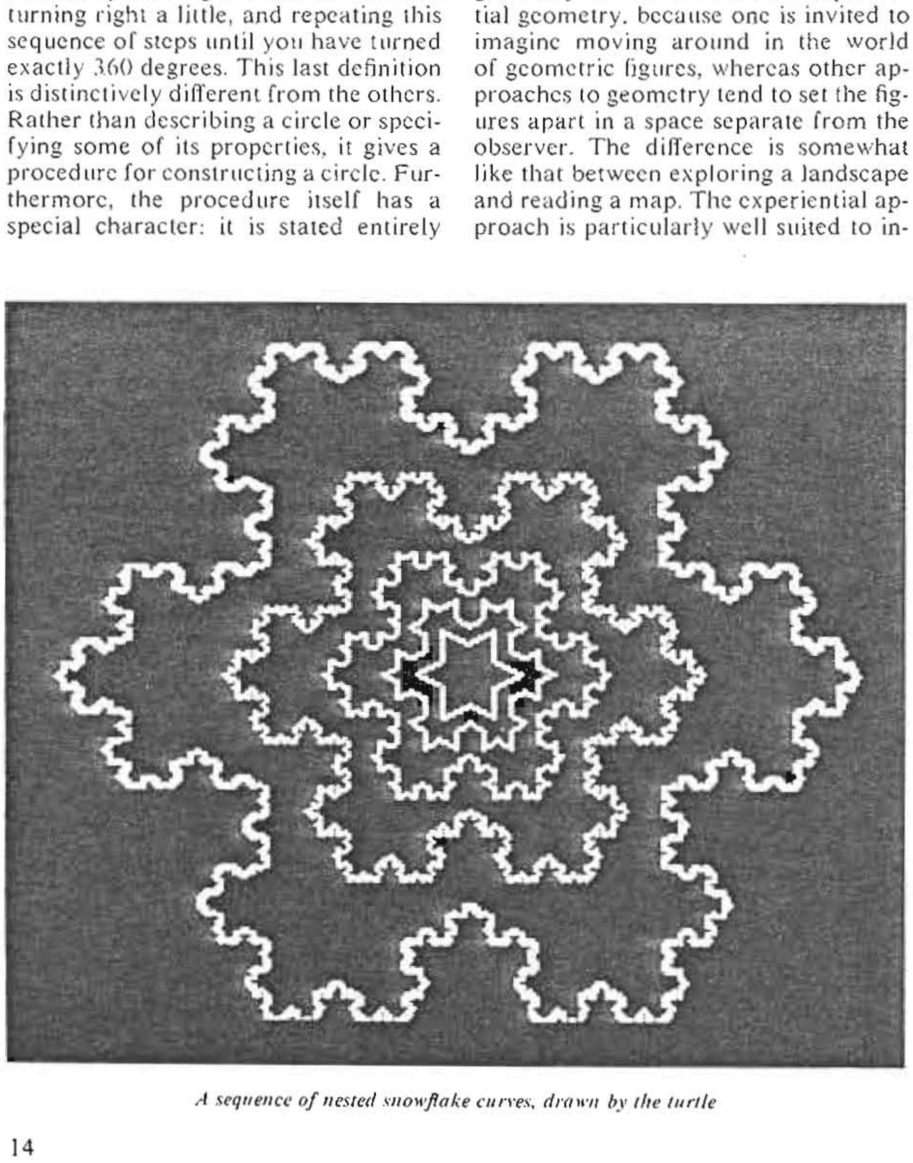
Turning turtle gives one a view of geometry from the inside out

by Brian Hayes

What is a circle? Is it the limiting case of an *n*-sided polygon as *n* goes to infinity? Is it the special case of an ellipse whose two foci coincide? Is it the locus of all points in a plane equidistant from a given point?

All these definitions are correct, of course, and more could be given. Consider this one: A circle is the figure formed by walking forward a little, then turning right a little, and repeating this sequence of steps until you have turned exactly 360 degrees. This last definition is distinctively different from the others. Rather than describing a circle or specifying some of its properties, it gives a procedure for constructing a circle. Furthermore, the procedure itself has a special character: it is stated entirely in terms of the "local" properties of the circle. The curve can be created by attending to one's immediate neighborhood only; there is no need for an overview. There is no need to know where the center of the circle lies, or even to determine the radius.

Procedures of this kind form the essence of a new way of thinking about geometry. It has been called experiential geometry, because one is invited to imagine moving around in the world of geometric figures, whereas other approaches to geometry tend to set the figures apart in a space separate from the observer. The difference is somewhat like that between exploring a landscape and reading a map. The experiential approach is particularly well suited to investigating geometric ideas with the aid of a computer. The procedural definitions are readily converted into computer programs, which can be executed to draw the shapes specified.



A sequence of nested snowflake curves, drawn by the turtle

The new way of thinking about geometry has come to be known as "turtle geometry." It is closely connected with the programming language Logo, which in turn has its roots in the Massachusetts Institute of Technology. Logo was conceived in the 1960's by Seymour Papert of M.I.T., primarily as a language for introducing children to computers. Many others have since contributed to its development and to its applications both in education and in other fields. Among them are Harold Abelson and Andrea A. diSessa of M.I.T., who have set forth the ideas underlying turtle geometry in a remarkable expository work: *Turtle Geometry: The Computer as a Medium for Exploring Mathematics.*

The original turtle was a mechanical device: a small wheeled vehicle whose movements could be controlled by instructions typed on a computer keyboard. The first such creature was built by the British neurophysiologist W. Grey Walter in the late 1940's. It had a dome-shaped cover somewhat like a turtle's shell. A mechanical turtle can move forward or backward and can change direction by pivoting in place. A pen can be mounted on the undercarriage, so that when the turtle is made to wander over a sheet of paper, it leaves a record of its path. Today such "floor turtles" are less common than "screen turtles," which move and draw on the surface of a cathode-ray tube. The turtle itself is represented on the screen by a simple triangular form, which moves in response to commands or programs entered at the keyboard.

Most methods of drawing with a computer employ a global system of coordinates, generally a Cartesian one, with two perpendicular axes. The position of every point on the screen is defined with respect to some origin, such as the lower left-hand corner, where the imaginary axes cross. Directions in the plane are also absolute. A command for drawing a line might be issued by specifying the two endpoints; for example, giving the points {0,0} and {100,0} might draw a horizontal line 100 units long across the bottom of the screen.

It should be emphasized that in any system of global coordinates the effect of a command does not depend on the sequence of commands that went before. The situation is quite different in a turtle-geometry system. A line 100 units long can be drawn by giving the turtle an instruction such as FORWARD 100. Where the line appears on the screen, however, and what its orientation is depend entirely on the state of the turtle at the time the command is issued. At any moment the turtle has both a position and a heading. The FORWARD 100 command causes it to proceed 100 units from its current position in the direction defined by its current heading (that is, in the direction it is "facing"). Hence the same command has a quite different result if it is given when the turtle is in a different initial state.

A simple turtle-geometry system can be created with just two commands: FORWARD and RIGHT. FORWARD is followed by a number that specifies how many "steps" the turtle is to take. RIGHT is also followed by a value, which gives the number of degrees the turtle is to turn right from its present heading. Any plane figure can be drawn by a sequence of forward moves and right turns. (Note that a 90-degree left turn can be specified in two ways: as RIGHT 270 or as RIGHT −90.) Nevertheless, a practical turtle system generally includes BACK or REVERSE and LEFT commands as well. PENUP and PENDOWN determine whether or not the turtle draws a line along its trail. A few other commands that report on the state of the turtle and offer a means of specifying absolute coordinates are also commonly provided, but they are not essential to turtle geometry.

The kind of geometry done with a turtle is formally called finite differential geometry. It is finite because the turtle can move only in discrete steps. It is differential because all movements are defined in terms of the difference between the present position and heading and the subsequent one. This is another way of saying it is a kind of geometry concerned only with the local properties of lines, curves and surfaces; there is no reference to distant points or to the global properties of a geometric figure. It follows that turtle geometry is most useful for exploring the "intrinsic" properties of geometric figures, those that are defined entirely within the figure.

One idea that can be awkward to formulate in a coordinate system but that comes forth clearly in turtle geometry is that of curvature. In Cartesian coordinates the curvature of a line in a plane might be defined as the rate of change in the slope of the line; the slope in turn can be defined as the rate of change in the y coordinate as a function of the x coordinate. The turtle can keep track of curvature in a much simpler way: it is the total turning per unit of distance traveled. Thus in a circle defined by repeating the instructions FORWARD 1, RIGHT 1 the curvature is everywhere equal to 1.

The concept of total turning leads to other interesting findings. Consider the familiar theorems of Euclidean geometry that give the sum of the interior angles of a convex polygon. For a triangle the sum is 180 degrees, for a quadrilateral 360, for a pentagon 540 and so on. In other words, the interior angle is invariably an integer multiple of 180 degrees, and the integer is equal to 2 less than the number of sides. The total turning of the turtle is measured in terms of a different angle: it is neither the interior nor the exterior angle but the change in heading at each vertex. Theorems about the total turning in a polygon can also be formulated. They are somewhat different from the theorems that pertain to the total interior angle, and they are also more general.

The most fundamental theorem states that the total turning when the turtle traces a polygon must be an integer multiple of 360 degrees. The proof is simple and transparent. If a figure is closed (as every polygon is), the turtle must eventually return to its starting point, and when it gets there, it must have precisely the same heading it had at the outset. (Indeed, this observation serves as an effective definition of geometric closure.) If the heading is the same, however, then the total turning must be zero degrees or 360 or some multiple of 360.

For a convex polygon a stronger result can be proved, namely that the total turning is exactly 360 degrees. (A polygon is convex if a line connecting any two points on its edges never passes outside it.) The theorem itself, however, applies to any polygon, including polygons that are not convex (such as a star-shaped pentagon). It applies even to closed figures that have curved edges and therefore are not polygons.

In coordinate geometry the total interior angle of a convex polygon has few obvious connections with other ideas. In turtle geometry, on the other hand, the concept of total turning is a powerful tool rich in wider applications. For example, Abelson and diSessa show that it can be used to analyze the topology of a closed path. A topologically simple path is one that can have any number of edges, curves and convolutions, provided only that it does not cross itself; like a simple convex polygon, it has a total turning of 360 degrees. Adding a single loop (and a single crossing) to the path makes the total turning either zero or 720 degrees, depending on the handedness, or direction, of the loop. Each additional loop adds or subtracts another increment of 360 degrees. Because of this property, the topology of the path can be determined from a turtle's-eye view of it, with only a microscopic section being visible at any one time.

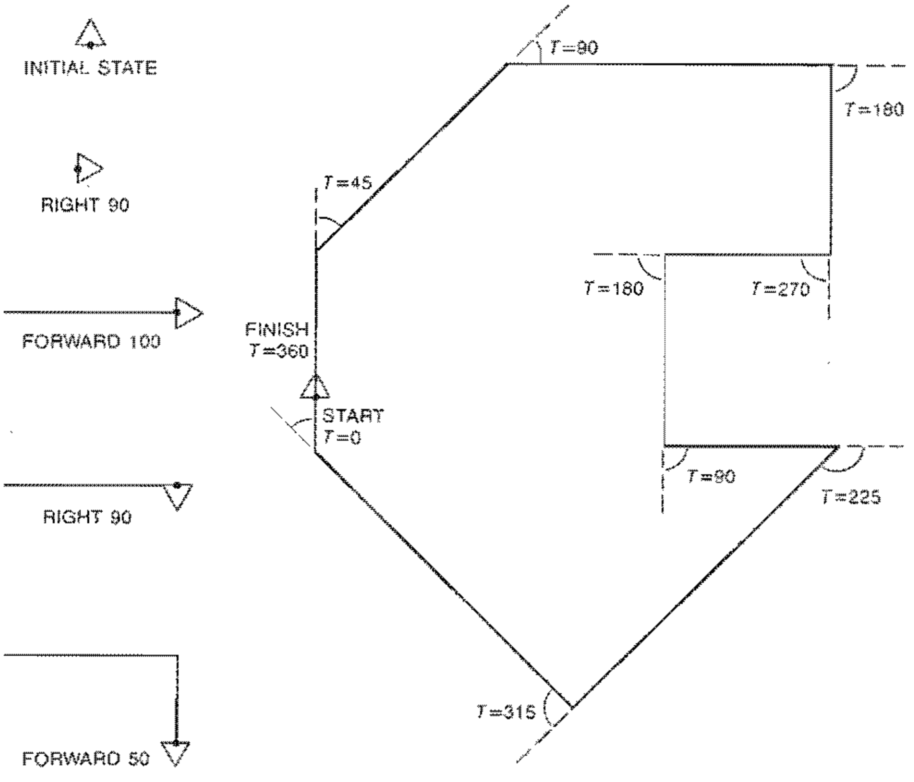
Total turning is also the basis of an algorithm that enables the turtle to find its way out of a maze. Abelson and diSessa state the algorithm as follows.

"1. Select an arbitrary initial direction, call it 'north,' and face that way.

"2. Walk straight 'northward' until you hit an obstacle.

"3. Turn left until that obstacle is on your right.

"4. Follow the obstacle around, keeping it on your right, until the total turn-



Some turtle commands (left) *and the concept of total turning* (right) ing (including the initial turn in step 3) is equal to zero.

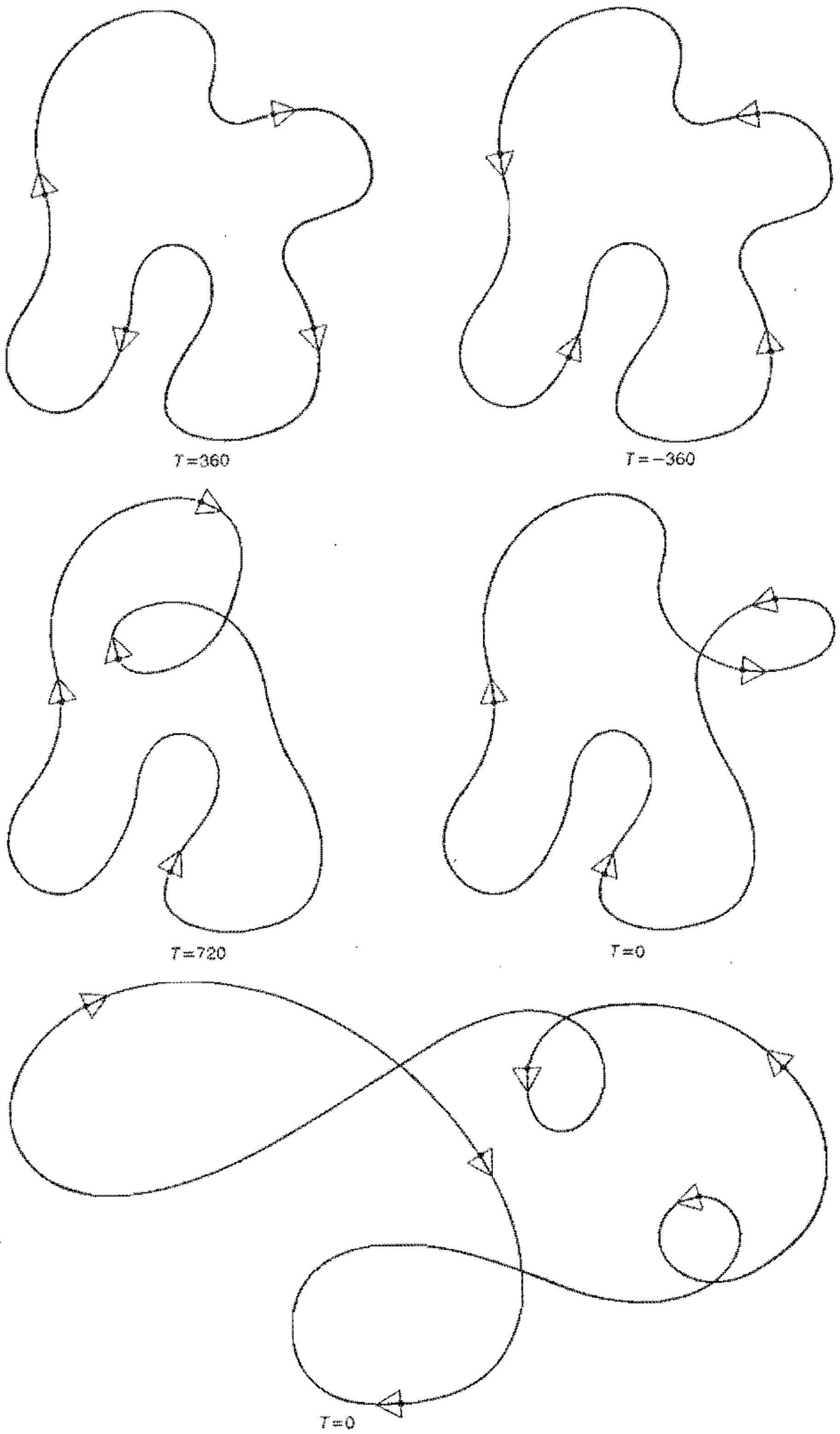
"5. Go back to step 2."

By this method the turtle can solve any fair maze (meaning one that actually has an exit). The procedure works because the only way to trap the turtle is to lead it into an endless loop, and keeping track of the total turning avoids that trap. Note that again a global problem—finding a route out of the maze—has been solved even though the turtle has information about only its local environment; there is no aerial view of the maze. The universal maze-solving procedure is called the Pledge algorithm, after John Pledge of Exeter in England, who developed it at age 12.

Still more applications of the total turning come up when the turtle is allowed to roam on surfaces that are not flat. On a sphere, for example, a closed path is found to have a total turning of less than 360 degrees. Furthermore, the amount of deviation from 360 degrees depends on the length of the path or, to be more precise, on the area it encloses. Readers familiar with the non-Euclidean geometries developed in the 19th century will recognize what is happening here: the deviation, or angular excess, is a measure of the curvature of the surface. In this instance it is important that the measurement can be made with only local and intrinsic information. The curvature one would most like to measure is that of the spacetime of the universe, and it will obviously have to be done by means of observations made within the universe. Abelson and diSessa explore this process in the final chapter of their book, which gives a turtle-geometric formulation of the general theory of relativity.

They also consider the meanderings of the turtle on the surface of a cube, which has the same topology as a sphere but a different geometry. Indeed, the geometry is bizarre. Suppose the cube is 100 units on a side and the turtle is initially in the center of a face and oriented parallel to an edge. Repeat the instructions FORWARD 100, RIGHT 90 three times. The result is a closed figure with three equal sides and three equal angles; it is also an equal-sided polygon whose vertexes are all right angles. Is it an equilateral right triangle or a three-sided square?

Another remarkable object of cubical geometry is the monogon, a one-sided polygon; it is a closed path formed by a turtle walk that has no turning at all. Even without a computer-driven turtle it is possible to see how a simple monogon can be formed: simply draw an "equator" on the cube. A more difficult question is this: Is there any combination of initial position and heading on the surface of the cube that does not close to form a monogon when the line is extended indefinitely? What if the heading must be specified in rational numbers? The answer will be found in *Turtle Geometry* (see chapter 6), but Abelson and diSessa would strongly discourage one from looking there. At one point they post the following warning: "DANGER—The next section contains 'premade' (already discovered) mathematics. It may be harmful to your imagination and should be used only as a last resort."



The topology of closed curves deduced from total turning

The discovery of a principle and the exploration of its extensions or generalizations are characteristic of the style in which turtle geometry is done. The computer greatly facilitates the process: experiments are easy, and variations on a theme can be tried with little effort. It is a geometry for tinkerers.

Consider the following Logo procedure, discussed by Abelson and diSessa and by other writers on the language:

```
TO SQUIRAL :DISTANCE
FORWARD :DISTANCE
RIGHT 90
SQUIRAL :DISTANCE + 5
END
```

Here SQUIRAL is the name of the procedure and DISTANCE is a variable whose initial value is to be given when the procedure is executed. (The colon is a Logo convention for identifying variables.) The turtle is instructed to move forward this amount and execute a 90-degree right turn; then the SQUIRAL procedure is invoked again, but with a larger DISTANCE value. The result is a "square spiral" that grows outward toward the edge of the screen. (In the procedure as it is given here the spiral goes on growing indefinitely, although only a finite part of it can be displayed.)

Many variations are possible. Changing the value of the constant that is added to DISTANCE each time the procedure is called merely alters the spacing between successive arms of the spiral. Multiplying by a constant value instead of adding one creates a spiral in which the arms get progressively farther apart in proportion to their distance from the center. Inserting a different constant angle in the RIGHT 90 instruction can convert the square spiral into a triangular or pentagonal or hexagonal one. An angle that differs only slightly from 90 degrees yields a set of nested "squares" that twist around their center, and so the vertexes form secondary spirals. A very small angle yields an approximation to a smooth "circular" spiral.

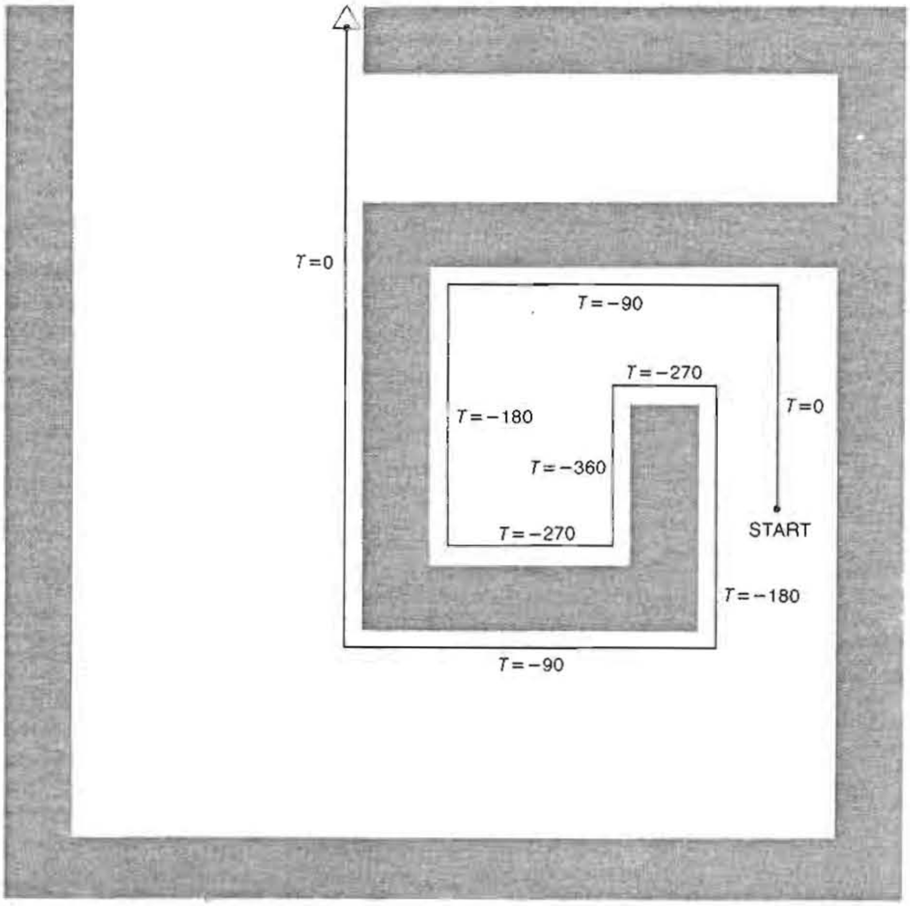
It is also possible to rewrite the procedure so that what changes with each invocation is the angle rather than the distance. The transformation is remarkable: instead of a single spiral that grows continuously outward the turtle draws an inward spiral and then an outward one of the opposite handedness, then another inward one and so on, creating a symmetrical array of spirals joined by their outermost loops. The underlying reason is that whereas distance can increase monotonically, angular measure is interpreted modulo 360, so that repeatedly adding a constant eventually returns the angle to a small value.

Consider only the subclass of curves formed when the initial angle is zero [*see illustration on page 20*]. All the patterns in this class are similar in basic form. The turtle creates a number of spirals of alternating handedness, then turns, retraces its path and after passing through its starting point creates another identical array rotated by 180 degrees with respect to the first one. Thereafter the turtle continues to retrace the same path indefinitely. The number of spirals in the pattern depends only on the angular increment, but the nature of the relation is not obvious. If the angular increment is not divisible by 8, the number of spirals is equal to one more than the largest factor of the increment that is not also a factor of 360. Can you perceive why? Can you predict the nature of the pattern when the increment is a multiple of 8?

It is the computer that makes possible such an exploratory approach to the family of spiral curves. Similar projects can be undertaken to explore the family of polygons, of tilings of the plane and of recursive figures such as the snowflake curve, which is made up of similar structures repeated at progressively smaller scales [*see illustration on page 14*]. The constructive, procedural nature of turtle geometry makes an important contribution to the process. It would be awkward at best to write an equation to specify the entire structure of the snowflake curve, whereas a procedure to generate the curve is readily broken down into a few elementary steps that are then carried out repeatedly.

Although turtle geometry and the Logo programming language have a strong historical connection, they are by no means inseparable. Abelson and diSessa note that turtles have been installed in at least two other languages (versions of APL and Smalltalk), and the programs in their own book are given not in Logo but in a related language they call Turtle Procedure Notation. They offer advice on the creation of a turtle system in BASIC and Pascal. A new language now being developed by members of the Logo group at M.I.T., called Boxer, also features a turtle.

Logo itself is a powerful general-purpose programming language. It has a strong kinship with Lisp, the list-processing language invented more than 20 years ago by John McCarthy. Indeed, the main differences between Lisp and Logo are matters of vocabulary and punctuation. The origin of Logo as a language for children has given it an anthropomorphic aspect that can be unsettling. The programmer writes as if he were speaking directly to the turtle, and occasionally the turtle talks back, issuing an error message such as "I don't know how to squiral." One should not be misled by this coyness into thinking the language is a plaything. The anthropomorphism is deliberate: it is part of a strategy to engage the programmer



The turtle, by following the Pledge algorithm, escapes from a simple maze (whether child or adult) in the turtle's experience of geometry. When one is baffled by a program, one is told that the answer is to "play turtle."

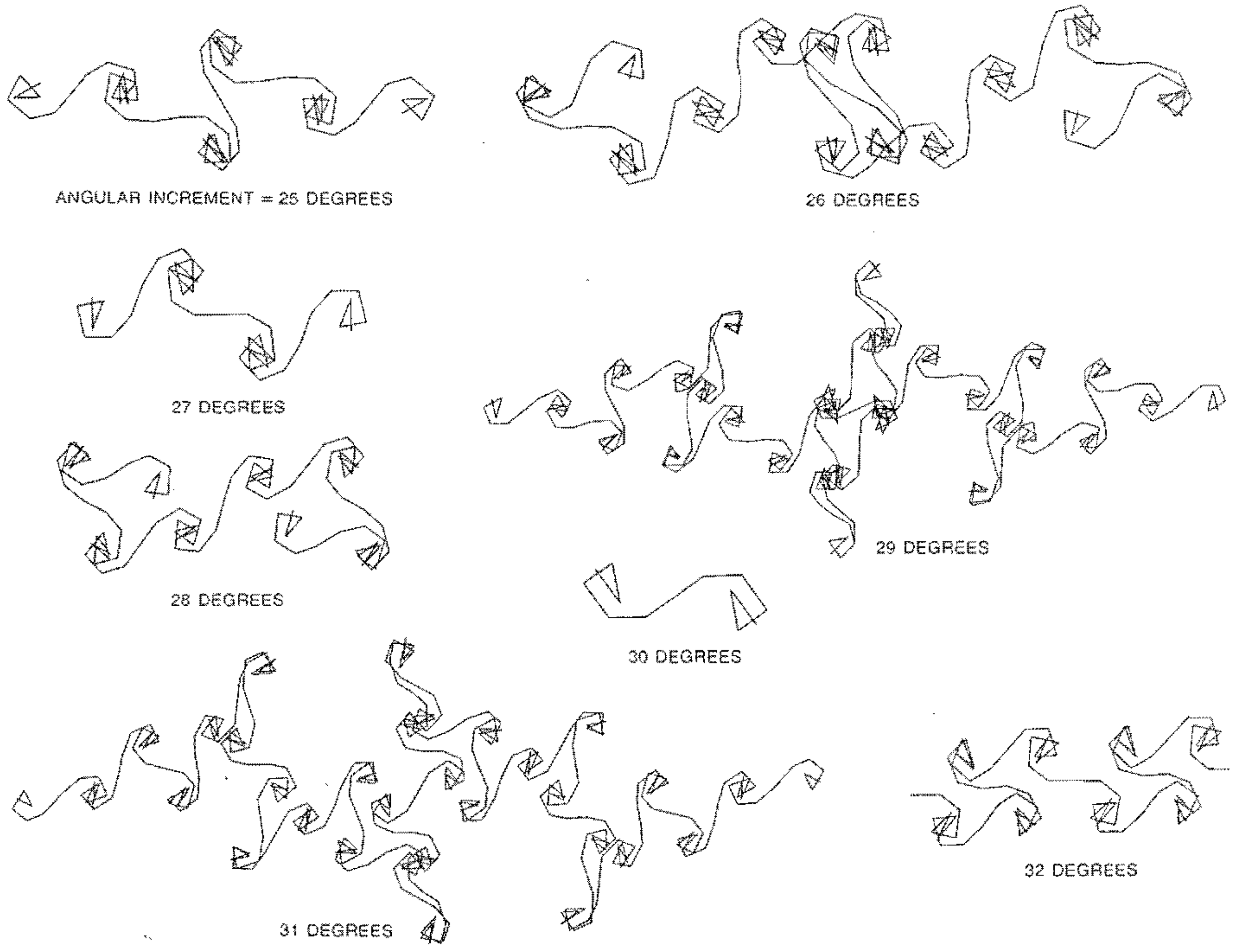
The deepest connection between turtle geometry and Logo is that they spring from a common philosophy of education. It is a philosophy based in large part on the work of Jean Piaget, which places the highest value on the student's own discoveries. Papert, who worked with Piaget for five years, declares his ambitions in *Mindstorms: Children, Computers, and Powerful Ideas.* "Programming the Turtle starts by making one reflect on how one does oneself what one would like the Turtle to do. Thus teaching the Turtle to act or to 'think' can lead one to reflect on one's own actions and thinking.... The experience can be heady: Thinking about thinking turns the child into an epistemologist, an experience not even shared by most adults."

The discussion of randomized prose in November elicited a number of comments I should like to pass on for the benefit of anyone considering a similar project. One algorithm I described called for searching through a text for each instance of a given sequence of characters, then building a frequency table for the letters that follow the sequence. The entire procedure was to be repeated for each letter of random text generated. Several readers proposed more efficient methods.

One approach, suggested by Judith E. Dayhoff and Stephen C. Locke, is to employ a data structure called a hash table. Each sequence of letters in the text can be taken to encode a numerical value, which can serve as an index pointing to an entry in the table. The entry gives the frequencies of the characters that follow the index sequence. Only sequences that actually appear need to be included. The procedure should be quite fast because the hash table is constructed once and subsequent references to it require only a calculation of the index, not a search of all the entries.

Bobby Bryant, James W. Butler, Ronald E. Diel, William P. Dunlap and Jim Schirmer pointed out still another algorithm that is not only faster than the one I gave but also appreciably simpler. It eliminates frequency tables entirely. When a letter is to be selected to follow a given sequence of characters, a random position in the text is chosen as the starting point for a serial search. Instead of tabulating all instances of the sequence, however, the search stops when the first instance is found, and the next character is the selected one. If the distribution of letter sequences throughout the text is reasonably uniform, the results should closely approximate those given by a frequency table.

An important historical precedent for work of this kind was brought to my attention by Sergei P. Kapitza, editor of *V Mire Nauki,* the Russian-language edition of Scientific American. The procedure for selecting a letter in the random-text program is known in probability theory as a Markov process, after the Russian mathematician Andrei A. Markov. Kapitza points out that Markov presented his first discussion of the process in terms of randomizing text. Markov's paper "On the Sequence of Letters in Eugene Onegin" asks to what extent Pushkin's poem remains Pushkin's when the letters are scrambled.



Eight arrays of spirals created by adding a fixed increment to an initial angle of zero