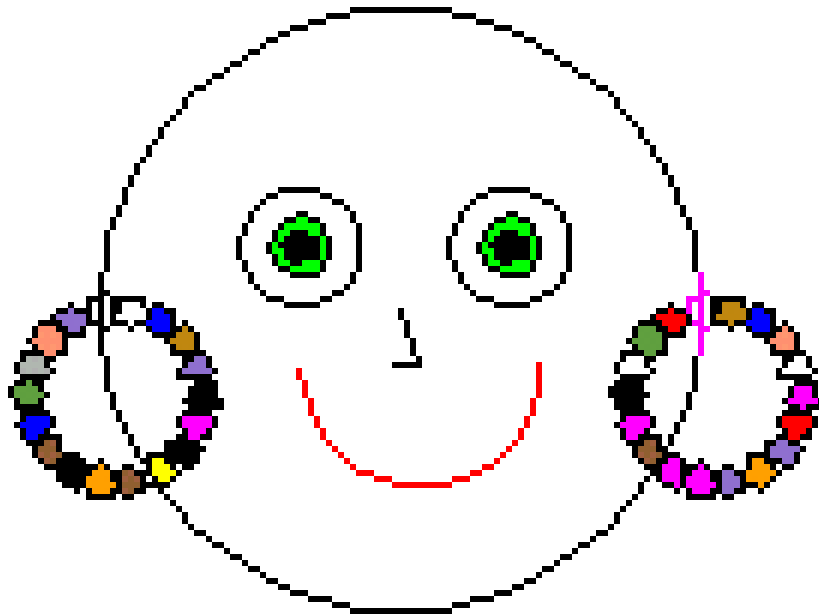


# Loca for Logo

Logo Designs, Lesson Plans, and Resources

by Erica T. Marciniac  
10.25.00 - Updated 7.16.03

TBE 540 California State University  
Dominguez Hills



# pinwheel

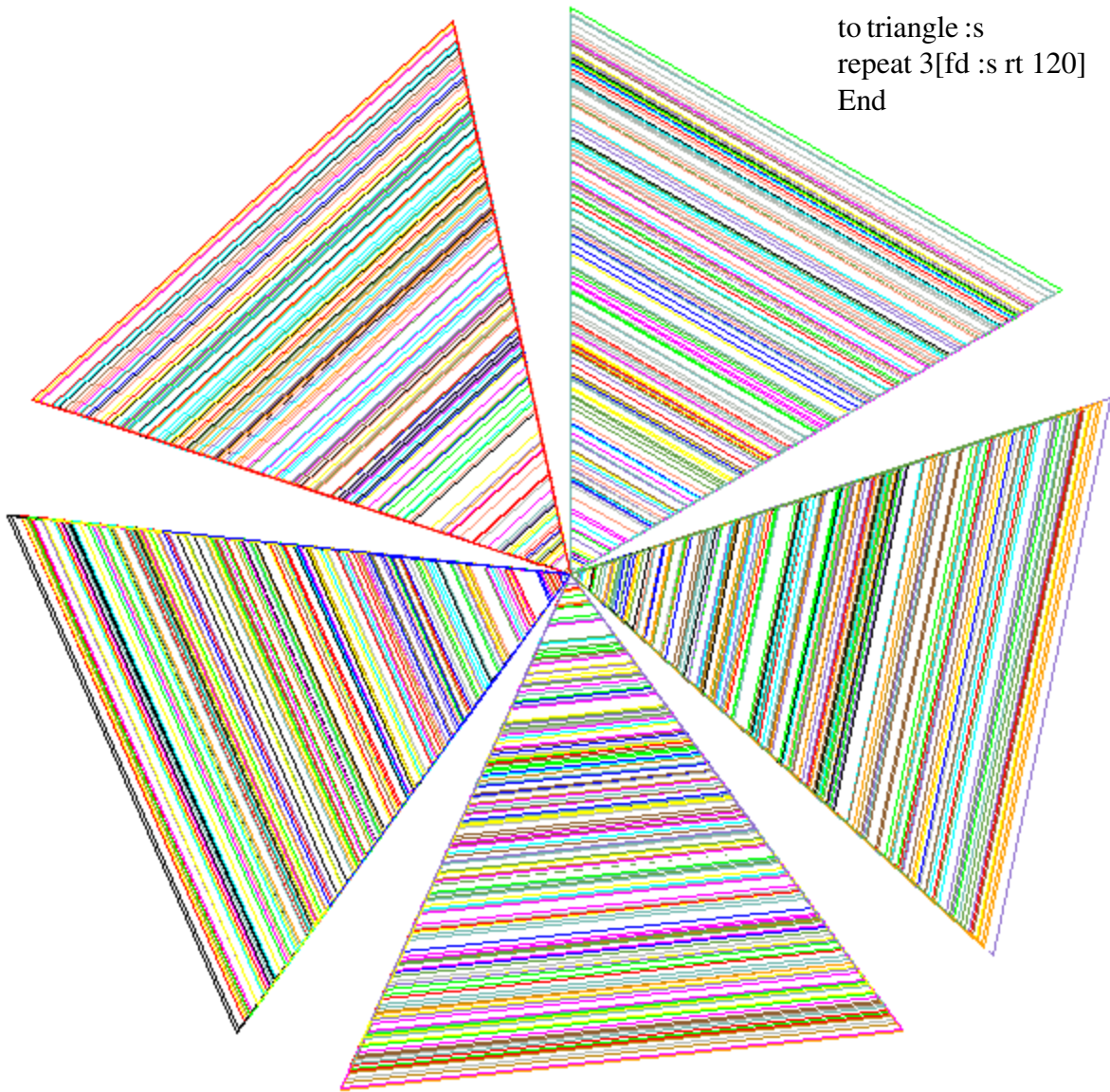
```
to pinwheel  
repeat 5 [rainbow.triangle lt 72]  
end
```

## Procedures Used:

```
to rainbow.triangle  
repeat 40 [pd triangles pu  
setpc 1 + random 257]  
end
```

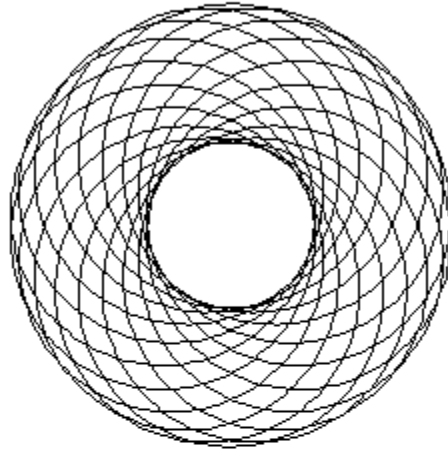
```
to triangles  
repeat 5 [triangle  
1 + random 300]  
end
```

```
to triangle :s  
repeat 3[fd :s rt 120]  
End
```



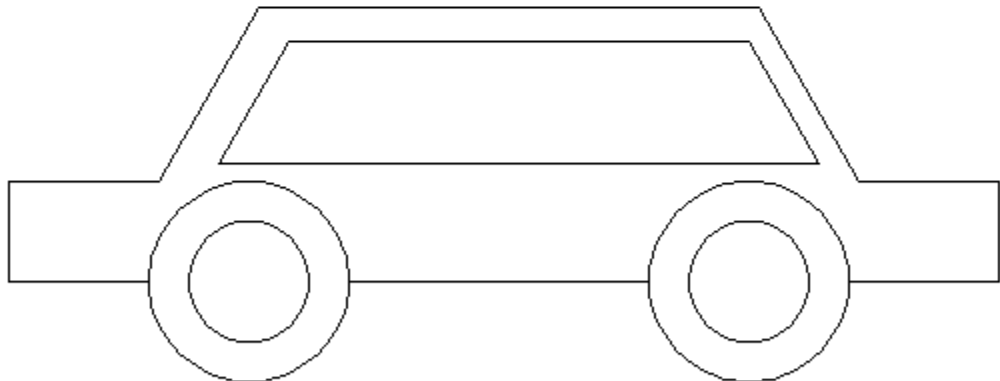
# design

```
to design
repeat 18[circle 76
lt 20 pu fd 12 pd]
end
```



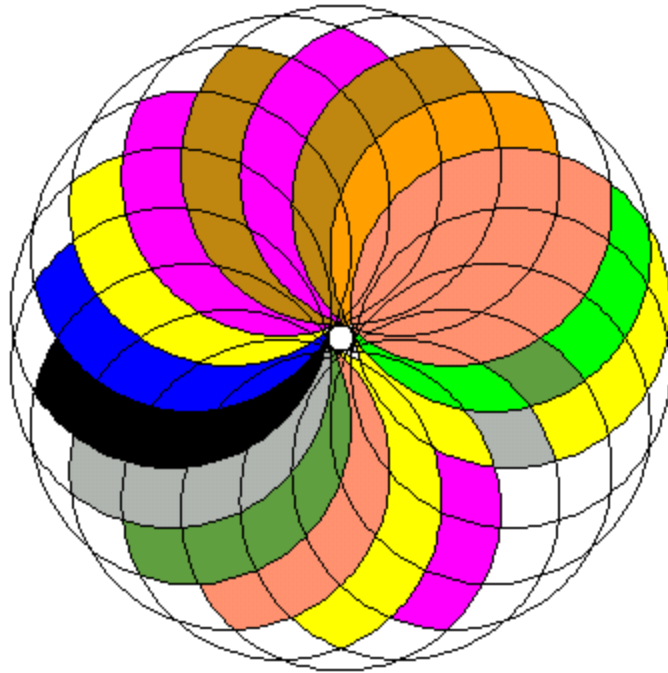
# car

```
to car
pd circle 50
rt 90 pu fd 50
pd
fd 150
pu fd 50 pd circle 50
pu fd 50 pd fd 75
lt 90 fd 50 lt 90 fd 70
rt 60 fd 100 lt 60 fd 250
lt 60 fd 100 rt 60 fd 75
lt 90 fd 50 lt 90 fd 70
pu fd 50 pd circle 30 pu fd 250 pd circle 30
lt 90 pu fd 120 lt 90 pd fd 230
lt 60 fd 70 lt 120 fd 300 lt 120 fd 70
end
```



# dots 80 30

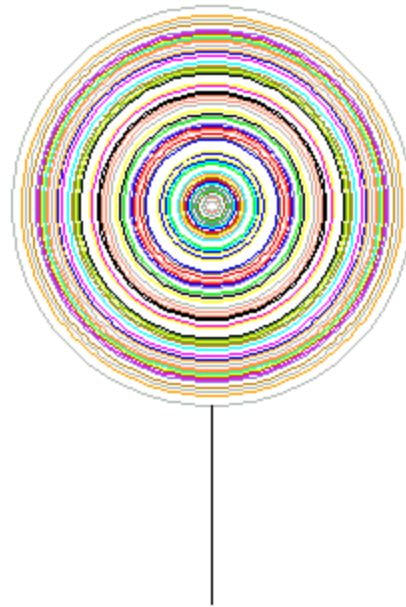
```
to dots :s :x
repeat 18[circle :s
setfc 1 + random 259
fill
lt 20 pu fd :x pd]
end
```



# bald.lady

```
to bald.lady
setpc 0
circle 50
lt 90 pu fd 50 pd dots 3 5 pu bk 100 pd dots 3 5
pu fd 50 pd setpc 0 lt 110 fd 10 rt 110 fd 5
pu bk 5 lt 90 pd setpc 4 arc 20
rt 90 pu fd 20 rt 90 fd 15 pd eye pu bk 35 pd eye
end
```

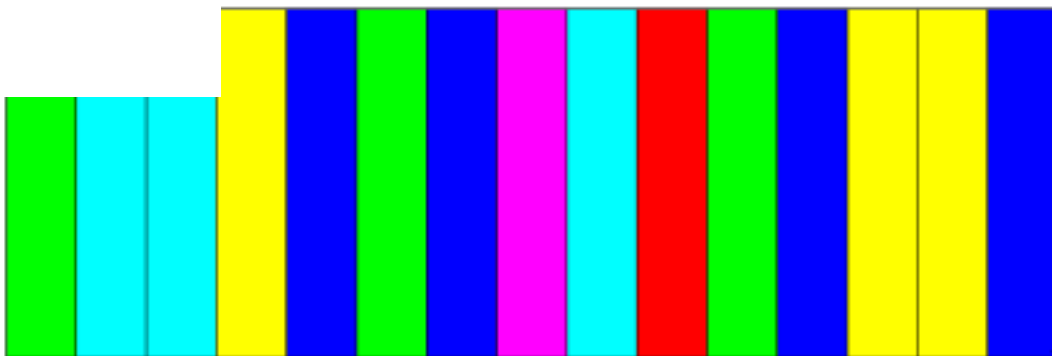
andom 256  
]



**xes**

50 50 rt 90 fd 50 lt 90]

ndom 6 fill pu fd 50 pd]



# flowers.everywhere

```
to flowers.everywhere  
  repeat 5[repeat 100[put.flower]]  
end
```

## Procedures Used:

```
to put.flower
```

```
  flower pu
```

```
  setx -500 + random 1000
```

```
  sety -500 + random 1000 pd
```

```
end
```

```
to flower
```

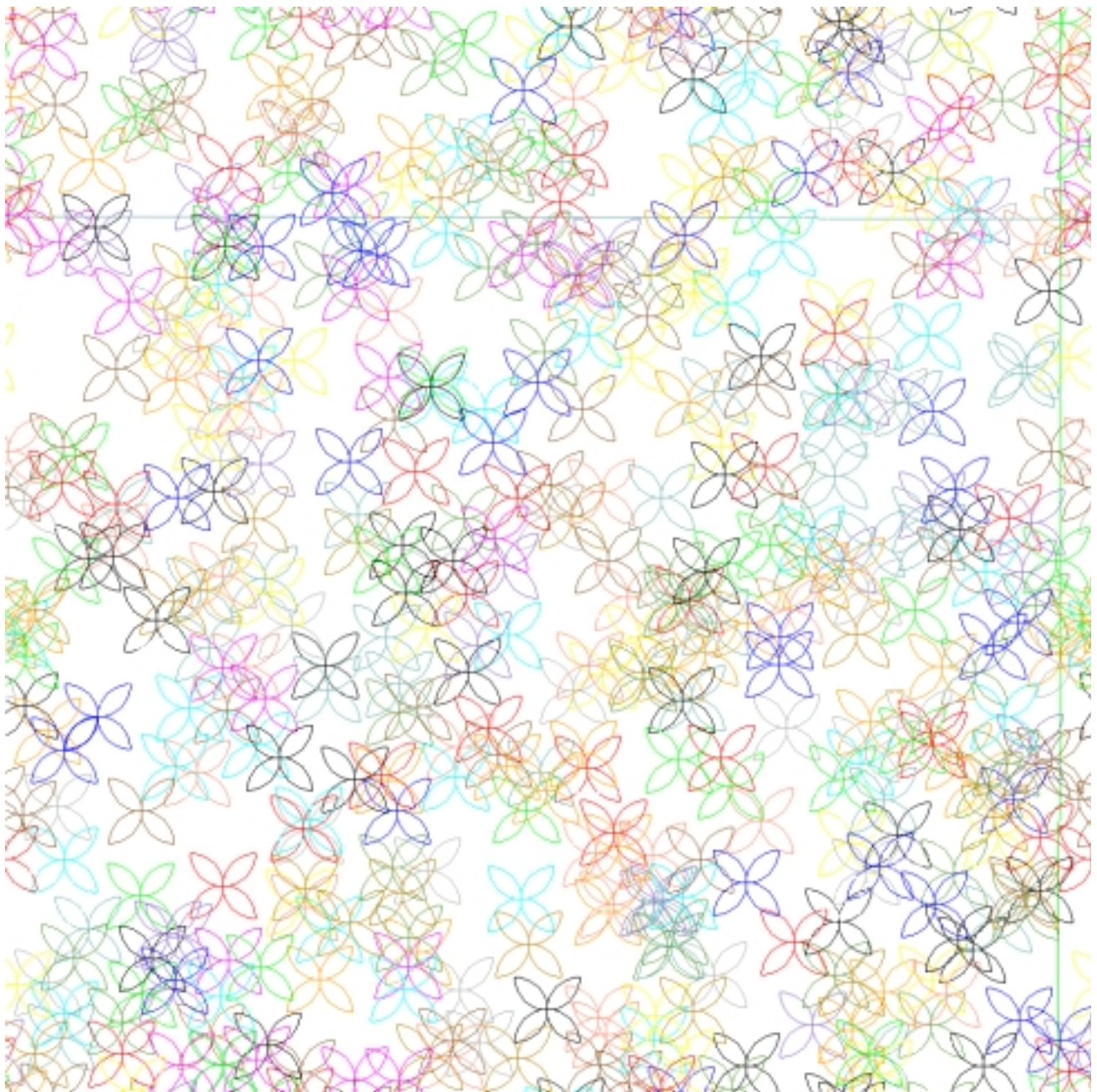
```
  setpc 1 + random 259
```

```
  arc 30 lt 90 pu fd 60 pd lt 180 arc 30
```

```
  pu fd 30 lt 90 fd 30 lt 90 pd arc 30
```

```
  lt 90 pu fd 60 pd lt 180 arc 30
```

```
end
```



# random.story

to random.story

make "noun [gargoyle troll potato dragon monster boy man]

make "verb1 [walking skipping jumping running]

make "verb2 [leaped jumped popped]

make "verb3 [grabbed tickled scared levitated]

make "adjective [goofy silly ugly gnarled slippery overdressed]

make "noun2 [market London CSUDH Inglewood]

make "word1 item (1 + random 6) :adjective

make "word2 item (1 + random 7) :noun

make "word3 item (1 + random 4) :verb1

make "word4 item (1 + random 4) :noun2

make "word5 item (1 + random 3) :verb2

make "word6 item (1 + random 4) :verb3

make "word7 item (1 + random 6) :adjective

make "word8 item (1 + random 7) :noun

pr (se "The :word1 :word2 "was :word3 "to :word4 "when "a :word7 :word8  
:word5 "out "and :word6 "him[.]

End

## repeat 10[random.story]

The slippery gargoyle was skipping to Inglewood when a overdressed gargoyle leaped out and grabbed him .

The slippery gargoyle was running to market when a goofy gargoyle jumped out and grabbed him .

The ugly troll was jumping to market when a slippery troll jumped out and tickled him .

The goofy man was skipping to London when a gnarled dragon popped out and scared him .

The gnarled monster was jumping to Inglewood when a goofy gargoyle popped out and scared him .

The overdressed gargoyle was jumping to London when a slippery monster popped out and tickled him .

The goofy potato was jumping to London when a slippery troll popped out and scared him .

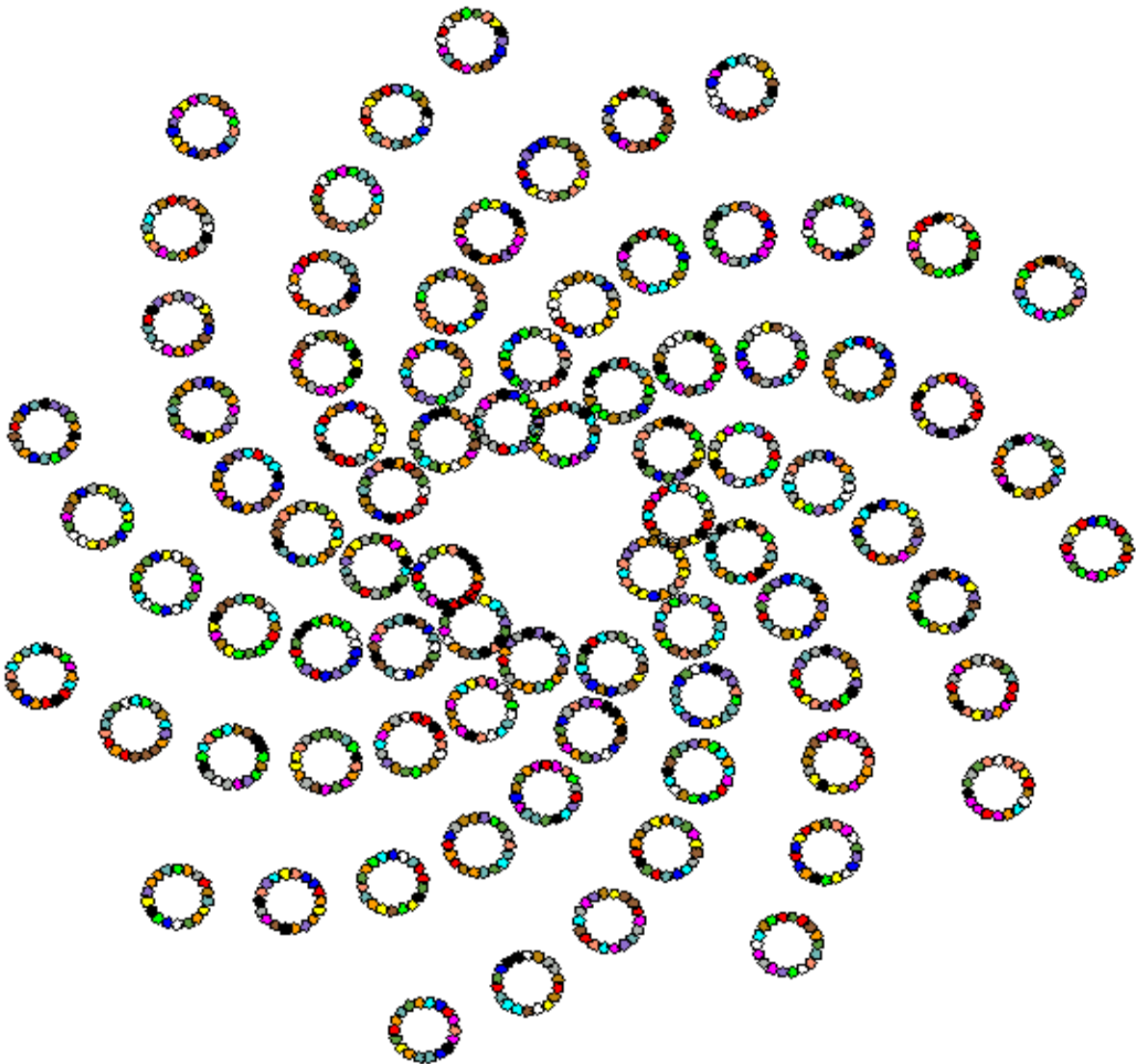
The overdressed man was jumping to Inglewood when a slippery troll popped out and scared him .

The goofy boy was walking to London when a ugly dragon popped out and scared him .

The slippery monster was walking to Inglewood when a slippery man jumped out and tickled him .

# recursion1

```
to recursion1 :s  
  if :s > 500 [stop]  
  pu fd :s pd dots 3 5 lt 130  
  recursion1 :s + 5 end
```



## Lesson: Dividing Circles into Equal Parts

**Software required:** MSW Logo for Windows

**Hardware required:** 1 computer for every 2 children, color printer

**Audience:** Students grades 4-5

**Curricular Integration:** Mathematics (geometry, fractions, division)

**Prerequisites:** Learners must already be able to...

- a. Operate a computer (basic knowledge)
- b. Given a simple shape, use the Logo commands below to make the shape:  
FD, BK, RT, LT, REPEAT, PU, PD.
- c. Create a procedure and be able to edit existing procedures.
- d. Use the POTS command to list procedures in memory.
- e. Save and load “workspaces.”

**Objectives:** After completing this lesson, the learner will be able to...

- a. Make a circle of any size radius.
- b. Use his/her knowledge of angles to divide a circle into equal parts.

**Lesson Description (time):**

1. Complete introductory activity page (procedure given; radius reviewed; equal parts) (15)
2. Teach (10)
  - a. Review procedure given; radius.
  - b. Discuss how they divided circle into thirds. How would you divide into fifths? Sevenths? (Divide 360 by # parts for angle size.)
3. In pairs, complete the follow-up activity—Dividing Circles into Equal Parts (20)

**Follow-up:** On homework pages and the unit test students will have some examples where they must divide circles into any given number of equal parts and state the angle.

# Dividing Circles Into Equal Parts

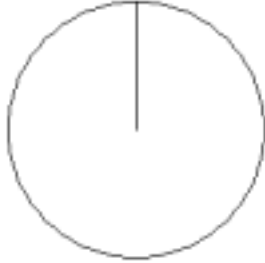
## Introductory Activity

Make this shape by typing the following commands:

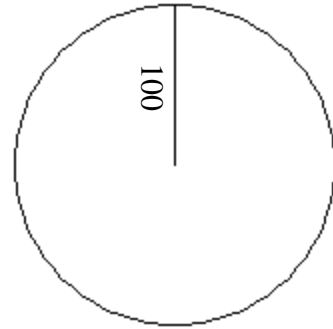
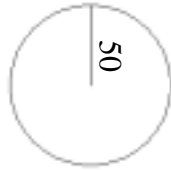
```
circle 80  
fd 80 bk 80
```

What is the radius  
of this shape?

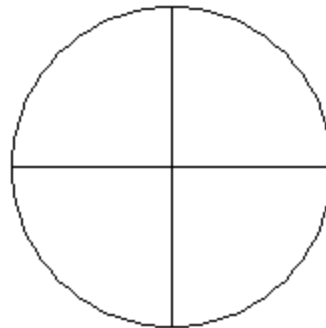
\_\_\_\_\_



Now make these two shapes:  
Write the steps you used next to the shapes.



Can you divide a circle into thirds and fourths? Try  
making these shapes.



# Dividing Circles Into Equal Parts

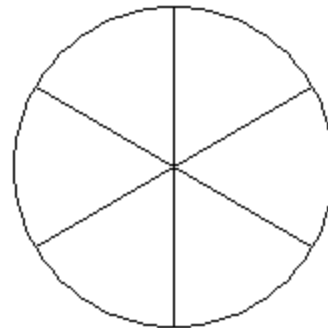
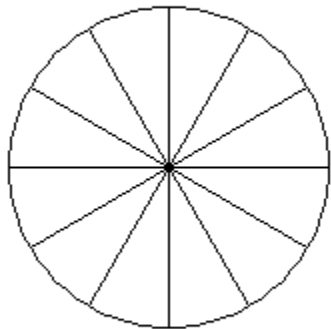
## Follow-up Activity

Here is the procedure for dividing a circle into thirds:

```
to thirds  
circle 80  
repeat 3[lt 120 fd 80 bk 80]  
end
```



Try modifying the procedure to make the following shapes:  
Write the steps you used next to the shapes.



Now make two more circles divided into equal parts--you decide how many parts.  
Draw the figure and write the steps you used below.

## Lesson: Making Circle Fractions Using SETFC and FILL

**Software required:** MSW Logo for Windows

**Hardware required:** 1 computer for every 2 children, color printer

**Audience:** Students grades 4-5

**Prerequisites:** Learners must already be able to..

- a. Operate a computer (basic knowledge)
- b. Given a simple shape, use the Logo commands below to make the shape:  
FD, BK, RT, LT, REPEAT, PU, PD.
- c. Create a procedure and be able to edit existing procedures.
- d. Use the POTS command to list procedures in memory.
- e. Save and load “workspaces.”
- f. Be able to edit a procedure to change circle size and divide a circle into a given number of equal parts.

**Objectives:** After completing this lesson, students will be able to...

- a. Make fractions with circles by filling one or more of the equal parts.
- b. Use commands SETFC and FILL and understand where to position the turtle to fill shapes.

**Curriculum Integration:** Mathematics (geometry, fractions, division)

**Lesson Description (time):**

1. Complete introductory activity (circle with different radii; different-sized equal parts; instructions for filling a circle) (10)
2. Teach: (20)
  - a. Go over ditto.
  - b. Teach and model filling a circle.
  - c. Teach/model filling an equal part in a procedure they have worked with (PU, move into open area, PD, FILL)
  - d. Teach and model SETFC
3. In pairs, complete the follow-up activity—Making Circle Fractions Using SETFC and Fill (25-30)

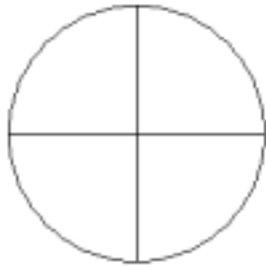
**Follow-up:** On homework pages and the unit test students will have some examples where they must make a fraction given a circle, stating the angle of each part.

# Making Circle Fractions Using SETFC and FILL

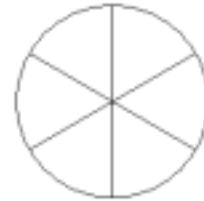
## Introductory Activity

Make these shapes and write the steps you used next to the shape.  
Notice that the radius size has changed!

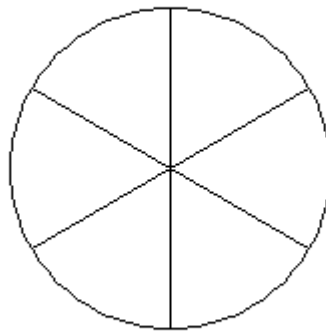
radius = 60



radius = 50



radius = 70



The command for filling a shape is FILL.  
To use this command, the turtle must be inside the shape.  
Try making a circle and then filling it.  
See if you can make any other shapes and then fill them.  
Always write down the steps you used.

circle 80  
fill



# Making Circle Fractions Using SETFC and FILL

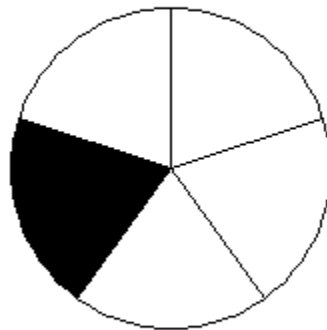
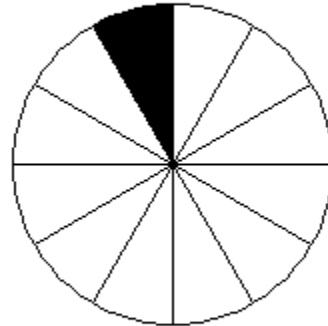
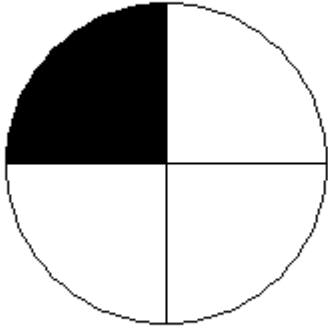
## Follow-up Activity

Try making the following shapes.

Write the steps you used **and** write the fraction next to the shape.

Remember, to FILL, the turtle must be inside the shape.

To put the turtle inside the shape without leaving a line, use PU and PD.



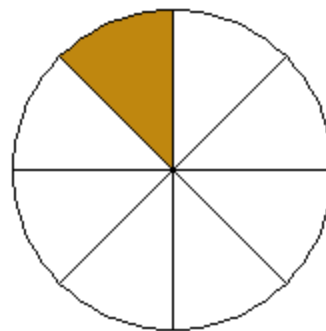
Now make your own fractions.

Set the fill color using the SETFC command.

Write your steps and write the fraction. (Use another sheet of paper if you need to.)

Here is an example:

```
to one.eighth  
circle 80  
repeat 8[fd 80 bk 80 lt 45]  
lt 30 pu fd 20 pd  
setfc 9  
fill  
end
```



## Lesson: Making Circle-Based Line Designs Using SETPC

**Software required:** MSW Logo for Windows

**Hardware required:** 1 computer for every 2 children, color printer

**Audience:** Students grades 4-5

**Prerequisites:** Learners must already be able to..

- a. Operate a computer (basic knowledge)
- b. Given a simple shape, use the Logo commands below to make the shape:  
FD, BK, RT, LT, REPEAT, PU, PD, SETFC, FILL.
- c. Create a procedure and be able to edit existing procedures.
- d. Use the POTS command to list procedures in memory.
- e. Save and load “workspaces.”
- f. Be able to edit a procedure to change circle size and divide a circle into a given number of equal parts.

**Objectives:** After completing this lesson, the learner will be able to...

- a. Make circle-based line designs using what he/she knows about angles and radius size.
- b. Use the SETPC command to create colorful designs.

**Curriculum Integration:** Mathematics (geometry, fractions, division)

**Lesson Description (time):**

1. Complete introductory activity page (given an example with its procedure, use procedure to recreate given circle design shapes; try SETPC command) (25)
2. Teach (10)
  - a. Go over ditto—discuss angles, radii, procedure.
  - b. Teach SETPC.
  - c. Model double-decker shape with different colors.
3. In pairs, complete the follow-up activity—Making Circle Based Line Designs Using SETPC (25)

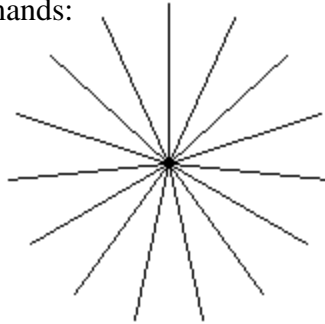
**Follow-up:** On homework pages and the unit test students will have some examples where they must indicate the angles in between lines of a given line design.

# Making Circle-Based Line Designs Using SETPC

## Introductory Activity

Make this shape by typing the following commands:

```
repeat 15[fd 80 bk 80 lt 24]  
end
```



What angle does the turtle turn  
before making each line?

\_\_\_\_\_

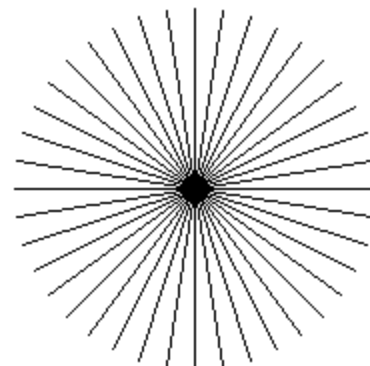
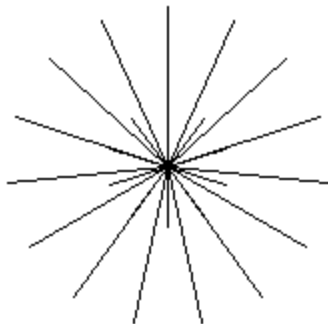
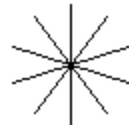
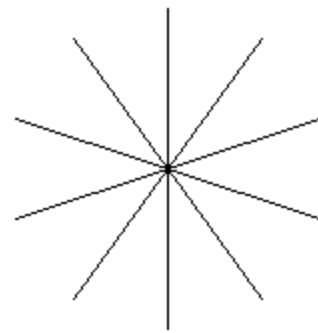
How many lines does the turtle make  
in this shape?

\_\_\_\_\_

What number do you get when you  
multiply the angle he turns by the  
number of lines?

\_\_\_\_\_

Use this information to help you make these shapes:  
Remember to write the steps you used.

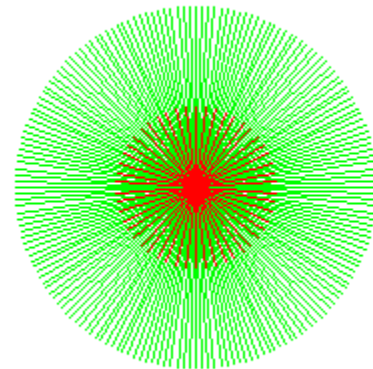
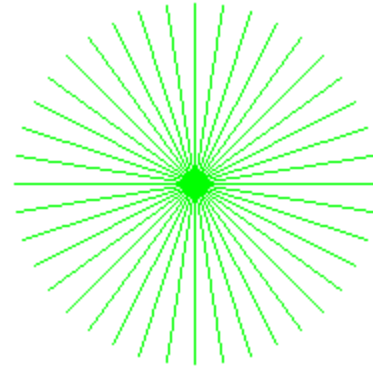
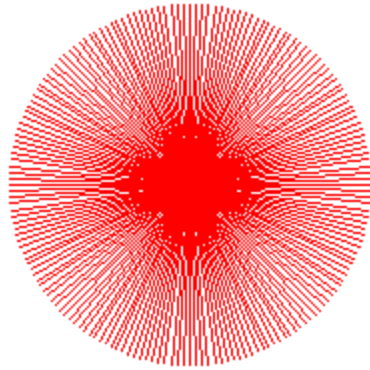


SETPC 4 changes the pen color to red.  
SETPC 2 changes the pen color to green.  
Can you find other colors, and then draw your  
design again?

# Making Circle-Based Line Designs Using SETPC

## Follow-up Activity

Try making these shapes:  
Don't forget to write your steps.



Now use SETPC to make your own circular designs. Record your steps and print out your pictures when you finish.

## Lesson: Using Variables and Procedures with Triangles

**Software required:** MSW Logo for Windows

**Hardware required:** 1 computer for every 2 teachers

**Audience:** teachers, any grade

**Prerequisites:** Learners must already be able to...

- a. Operate a computer (basic knowledge)
- b. Given a simple shape, use the Logo commands below to make the shape:  
FD, BK, RT, LT, REPEAT, PU, PD.
- c. Create and save procedures.
- d. Be able to edit existing procedures.
- e. Use the POTS command to list procedures in memory.
- f. Save and load “workspaces,” including previously learned triangle.

**Objectives:** After completing this lesson, learner will be able to:

- a. Create a procedure with a variable to make any size of a triangle.
- b. Create a procedure with a random variable.
- c. Build upon procedures to make a complex design.

**Lesson Description (time):**

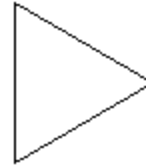
1. Complete introductory activity (load workspace; make different sizes of triangles; try using variable in procedure) (15)
2. Teach & model: (20)
  - a. How to make triangles with variable side lengths; save procedure.
  - b. How to make triangles with random variable side lengths; save procedure.
  - c. Repeat triangle with random variable; save procedure.
3. In pairs, complete hands-on exploration. (building on procedures to make pinwheel.) (25)

# Using Variables and Procedures with Triangles

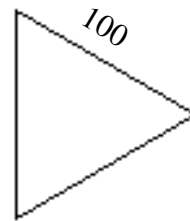
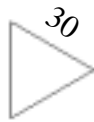
## Introductory Activity

This triangle was created using the following procedure:

```
to triangle
repeat 3[fd 80 rt 120]
end
```



Edit this procedure to make the following triangles:



To make a triangle with a side of variable length, use the following procedure:

```
to triangle :x
repeat 3[fd :x rt 120]
end
```

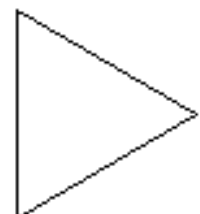
Save the procedure as triangle.  
Now, anytime you want to make a triangle of a particular size, tell the turtle what size triangle you want, as in the examples to the right.

triangle 30



Try making triangles of variable sizes.

triangle 100



Can you write and save a procedure for a square of variable size?

# Using Variables and Procedures with Triangles

## Hands-On Exploration

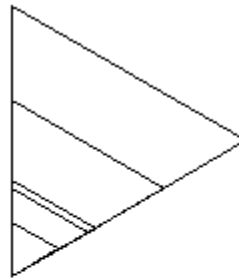
1. Using your triangle procedure, make a new procedure that defines the side of the triangle as a random number between 1 and 150 and call it `random.triangle`. Remember that for a random variable equal to or greater than 1, we write the variable: `1 + random x`, where `x` = the largest number you want. You must set a value for `x`.

```
to random.triangle  
triangle 1 + random 150
```

Draw a few `random.triangles` in Logo. Are they different sizes every time? They should be.

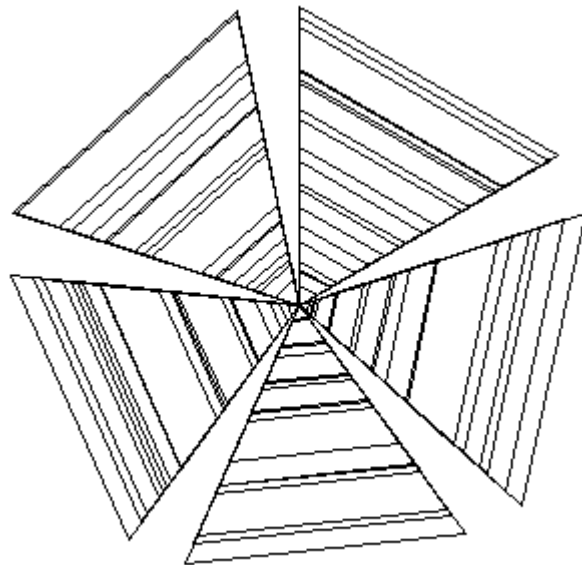
2. Now, write a procedure to REPEAT your `random.triangle` 5 times. Call this procedure `striped.triangle`. It should look like this:

or this:



Can you edit `striped.triangle` so that it has more than 5 stripes?  
Record your work.

3. Last, use the procedures you have already written to make a pinwheel. Save the procedure as `pinwheel`. Your pinwheel should look something like this: (number of stripes may vary)



*Possible solution for pinwheel:*

```
to pinwheel  
repeat 5[striped.triangle lt 72]
```

# Annotated Bibliography of Logo Resources

<http://www.edu.uleth.ca/students/logo/default.html>

This website is a comprehensive introductory guide to logo for teachers. It explains what logo is, the origins of logo, and how it can be used in the classroom. The site has links to twelve lesson plans for the K-3 teachers, as well as teaching tips and a nice glossary of logo commands. Additional references and software links are provided.

<http://www.kidsandcomputers.com/SiteToc.cfm>

This website is a first person account of how a dad and his then 4-year old daughter explored MicroWorlds logo, from first opening the box through creating some advanced designs. “Dad” introduces himself as a computer consultant working on a variety of contracts, and while the site is informational, it is obviously an advertisement as you can order MicroWorlds logo with your VISA card at the bottom of the page. Regardless, Dad argues that you can start teaching your kids to program at age 4—and he argues it quite convincingly...

<http://el.media.mit.edu/logo-foundation/>

This is the Logo Foundation’s website. They are a nonprofit organization offering “a place to find information and resources for learning and teaching logo.” The Logo Foundation is located in New York City and has many partner organizations including the MIT Media Lab and the Global SchoolNet Foundation. The site has links to just about anything you need related to logo: a calendar of logo seminars and events, chatrooms for logo users and teachers, reference links to other logo sites, links to logo products, a logo users’ newsletter, and links to the latest articles and publications about logo.

<http://www.math.utah.edu/~clemens/relearning.html>

This very interesting site from Monument Valley High School, Utah, gives information about a yearly project done by students at that school called ‘NDAHOO’ AAH. The project links logo programming to traditional Navajo designs. If you go to the [here](#) link on the third line of this page you get to directory which includes a program overview, short bios of Navajo storytellers and links to their stories, Navajo designs rendered in logo, lesson plans, and a list of resources. The designs are beautiful! What an excellent way to incorporate logo into the curriculum!

<http://mckoss.com/logo/>

Another “dad” site, this is from a man who started teaching logo to his son’s second grade class in 1995. The site is concise and good. He explains where to get a free version of logo and goes right into a series of introductory lessons. Especially interesting is the lesson on procedures, which he calls “making new words”—a nice way to put the concept for children. Another nice lesson is “making music with logo.” All lessons come with a worksheet where he provides the text and students use it to make their shapes. At the bottom of the page there is a link to “fun shapes” which include the code to make pumpkins and trees, etc.

<http://members.atlantic.net/~caggiano/logo/index.html>

This large website provides a host of logo information, links, and resources including MicroWorlds projects you can run from your browser, books, and information for teachers, parents, and kids. In the “screen shots” section, there are a variety of advanced logo designs. There is a page for online logo lessons that is under construction and apparently aimed at adults. Two schools are linked to the site. One has a series of links to completed logo projects; the other has links to many logo-related resources, as well as a place to create your own games. There are links to other logo sites, software resources, and a chatroom. The site is a member of the Logo Users Ring and teachers.net.

<http://library.thinkquest.org/18446/>

At this Logo website you can learn to program logo in English, French, or Dutch. Programs can be run from the website with a java download, which is provided. There is a nice series of 10 lessons in “speaking the logo language” and a logo workspace at the bottom of the page. More advanced users can go straight into making their own logo designs in the workspace. Teaching tips are provided as to how to organize the ten lessons into units. The site also features a monthly contest for logo programmers. There is a chatroom for site users and a list of procedures which can be used in conjunction with lesson #9 (about procedures.) Finally, the site ends with an “after logo” explanation of logo’s relationship to the programming world. A great site!

<http://moon.pepperdine.edu/~gstager/logoexchange/>

Part of Gary S. Stager’s Educational Services and Resources for Progressive Educators, the *Logo Exchange* is a journal of the ISTE’s Special Interest Group for “logo using educators.” You should be able to view the latest version of the journal online (although I couldn’t get it to work) as well as subscribe to it. According to the index, the journal contains articles relative to logo, as well as a teacher feature column and book reviews.

<http://el.www.media.mit.edu/groups/el/projects/starlogo/>

This Starlogo site allows you to download this specialized version of logo, which is used to mimic nature patterns—patterns with no centralized organization, from which patterns emerge nonetheless. As it states, “with starlogo, you can model and gain insight into many real-life phenomena, such as bird flocks, traffic jams, ant colonies,” etc. In the “sample projects” link, thumbnails of logo-produced designs are organized into scientific categories including biology, graphics, math, physics, and social systems—making it easy for the educator shopping for curricular applications. There are links to researchers and teachers using logo, which provide additional resources and examples of the multiple uses of Starlogo.

Brna, Paul. (1989). Programmed Rockets: An Analysis of Students’ Strategies. *British Journal of Educational Technology*, 20 (1), 27-40.

This article describes a computer simulation designed to examine secondary school students’ strategies in solving a physics problem involving the velocity of a rocket. Students’ beliefs about dynamics are discussed, and the LOGO programming language is used to explore the idea of velocity. Ways in which simulations can support teachers’ diagnostic powers are discussed.

Durnin, Robin. (1995). Computers and Clarifying Mathematical Thinking. *Kamehameha Journal of Education*, 6 (1), 63-68.

This article presents three computer activities that can help students learn mathematics concepts and clarify their thinking: LOGO programming, creating presentations to teach the solution of certain mathematical problems to other students in the class, and writing in order to explain their mathematical thinking. The activities provide immediate feedback, deepen understanding, and allow students to visualize concepts.

Lee, Mi Ok C. & Thompson, Ann. (1997). Guided Instruction in Logo Programming and the Development of Cognitive Monitoring Strategies Among College Students. *Journal of Educational Computing Research*, 16 (2), 125-44.

This article examines whether an approach to teaching Logo programming that directly guides college students in the use of cognitive monitoring skills and the transfer of those skills leads to increased cognitive monitoring and problem-solving skills. It demonstrates that guided instruction led to increased comprehension monitoring and contributed to the development of Logo error identification and debugging skills.

Lehrer, Richard & Randle, Lynn. (1987). Problem Solving, Metacognition and Composition: The Effects of Interactive Software for First-Grade Children. *Journal of Educational Computing Research*, 3 (4), 409-27.

This experimental study compared the instructional effectiveness of Logo programming, commercially available software designed to aid composition and problem solving, and traditional teaching methods for low socioeconomic status first-grade students. Both software environments enhanced problem solving performance for a novel task, but Logo was most facilitative for “learning to learn.”

Muscat, Jean-Paul. (1992). Polygons and Stars. *Mathematics in School*, 21 (2), 25-28.

This article explores the close relationship between pattern formation and traditional geometry using the Logo programming language with the specific example of joining squares, corner to corner, to form a closed ring. It includes the Logo programs utilized, as well as color illustrations of the interesting and eye-catching patterns generated.