

## > Project Summary and Outcomes

The Chaos Game provides a simple yet beautiful example of fractals, geometric shapes that have self-similarity (they look the same at any level of detail) and a fractional (non-integer) dimension. There will be three activities in this project. First, for the Playerless Chaos Game, students will implement an iterative algorithm to draw a fractal. Next, for the Sierpinski triangle activity, students will implement a completely different recursive algorithm that surprisingly draws the same shape. Finally, in the Chaos Game, students will implement a brain-teaser game based on the first two activities. (Note that Activities 2 and 3 are optional.)

In addition to introducing students to fractals, these three activities can also be used to introduce chaos theory, the study of the ability of a system to make long-term predictions from input parameters. Chaotic systems are not random. A classic application of chaos theory is in weather prediction, where precise, long-term predictions are difficult to make. Fractals are mathematical entities that model chaotic behavior. Students may want to explore even deeper, related issues, such as determinism compared with free will.

## > Student Assessment (see rubric)

**How well do students perform each of the following tasks:**

- Use good programming techniques to develop and test classes
- Implement algorithms to roll dice
- Draw equilateral triangles
- Draw points at proper places based on the Playerless Chaos Game
- Research and write about Sierpinski and the Sierpinski triangle

## > Prerequisite Knowledge and Skills

### Math:

- Basic trigonometry
- Midpoint calculation between two points
- Midpoint calculation for a line segment

### Java:

- Basic control structures
- Basic graphics (methods of the java.awt.Graphics class)

## > This Project Targets the Following Subject Areas(s):

Pre-Java	Java Programming
<input type="checkbox"/> Hardware Basics	<input checked="" type="checkbox"/> Applet Programming
<input checked="" type="checkbox"/> Software Basics	<input type="checkbox"/> Subroutine Programming
<input type="checkbox"/> Networks and Servers	<input type="checkbox"/> Full Scale Programming
<input type="checkbox"/> HTML	
<input type="checkbox"/> Action Scripting	
<input type="checkbox"/> Java Scripting	

## > Project-Framing Questions

<b>Essential Question</b>	<ul style="list-style-type: none"> <li>• How do observations of the natural world inform the study of mathematics, science, and technology?</li> </ul>
<b>Activity Questions</b>	<ul style="list-style-type: none"> <li>• What is chaos? What is a fractional dimension? What is a fractal?</li> <li>• Who was Sierpinski and how does his work relate to chaos and fractals?</li> </ul>
<b>Sample Content Questions</b>	<ul style="list-style-type: none"> <li>• How can we develop classes to implement the Playerless Chaos Game?</li> <li>• How can we develop classes to implement the Sierpinski triangle?</li> </ul>

## > Targeted Content Standards, Benchmarks, or State Frameworks

### Massachusetts Curriculum Frameworks:

#### Geometry:

**10.G.7** – Using rectangular coordinates, calculate midpoints of segments, slopes of lines and segments, and distances between two points (for example, by using the point-slope form of the equation).

**12.G.5** – Apply properties of angles, parallel lines, arcs, radii,

#### School-to-Career Competencies:

Reasoning  
 Making decisions  
 Thinking creatively  
 Solving problems  
 Seeing things in the mind's eye  
 Knowing how to learn

#### International Society of Technology Education (ISTE):

TL-IV.A – Apply technology in assessing student learning of subject matter using a variety of assessment techniques.

chords, tangents, and secants to solve problems.

## > Materials and Resources Required for Project

<b>Equipment</b>	<ul style="list-style-type: none"> <li>• Computer for each student with Java SDK on it</li> </ul>
<b>Consumable Supplies</b>	<ul style="list-style-type: none"> <li>• Diskettes or CDs</li> <li>• Graph paper</li> </ul>
<b>Textbooks/Lesson Guides</b>	<ul style="list-style-type: none"> <li>• See any of the many possible reference guides to basic Java, for example, <i>Java Software Solutions, AP Version</i> by John Lewis, William Loftus, and Cara Cocking. Addison Wesley Longman, 2003.</li> </ul>
<b>Technology</b>	<ul style="list-style-type: none"> <li>• Protractor</li> <li>• Compass</li> </ul>
<b>Internet Resources</b>	<ul style="list-style-type: none"> <li>• <a href="http://math.rice.edu/~lanius/frac/jurassic/jurassic.html">http://math.rice.edu/~lanius/frac/jurassic/jurassic.html</a> – for background on fractals</li> <li>• <a href="http://java.sun.com/j2se/1.4.2/docs/api/">http://java.sun.com/j2se/1.4.2/docs/api/</a> - select the <code>java.awt.geom</code> package for specifications on <code>Point2D.Double</code> class</li> <li>• <a href="http://www.imho.com/grae/chaos/chaos.html">http://www.imho.com/grae/chaos/chaos.html</a> – for the history of chaos theory</li> <li>• <a href="http://mathworld.wolfram.com/SierpinskiSieve.html">http://mathworld.wolfram.com/SierpinskiSieve.html</a> - for the history of Sierpinski</li> </ul>

## > Activity/Lesson Plan Outline

### Activity 1: The Playerless Chaos Game

**Overview:** For this activity, students will write an applet to draw the famous Chaos Game. In the first activity of the project, students will not yet build the game, but they will learn how a simple algorithm can draw a fractal. A good place to learn more about fractals is (<http://math.rice.edu/~lanius/frac/jurassic/jurassic.html>).

**Purpose:** To practice with simple Java control structures while implementing a simple yet beautiful algorithm that draws a simple fractal.

**Assessment:** How well do students design and implement the algorithm? How well do the students research and write answers to cross-curricular questions? See the attached rubric for details.

Here's how the game works: First, in an applet window, draw three points (pixels) that are the vertices of an equilateral triangle. Call the points  $p_1$ ,  $p_2$ , and  $p_3$ . Draw  $p_1$  in red,  $p_2$  in blue, and  $p_3$  in green. Now start at  $p_1$  and repeat the following a number of times:

- Roll a three-sided die. Let  $i$  be the value of the roll.
- Find the point that is halfway between the current point and  $p_i$ . Make this point the new current point and draw it in the same color as  $p_i$  (that is, if heading toward  $p_1$ , draw in red;  $p_2$ , draw in blue; or  $p_3$ , draw in green).

Start by iterating 100 times, then try 1000, 10,000, and 100,000 iterations. Keep your drawings centered in the applet panel. If the algorithm is implemented properly, all the points will be drawn inside the triangle formed by  $p_1$ ,  $p_2$ , and  $p_3$ .

## > Activity/Lesson Plan Outline (cont.)

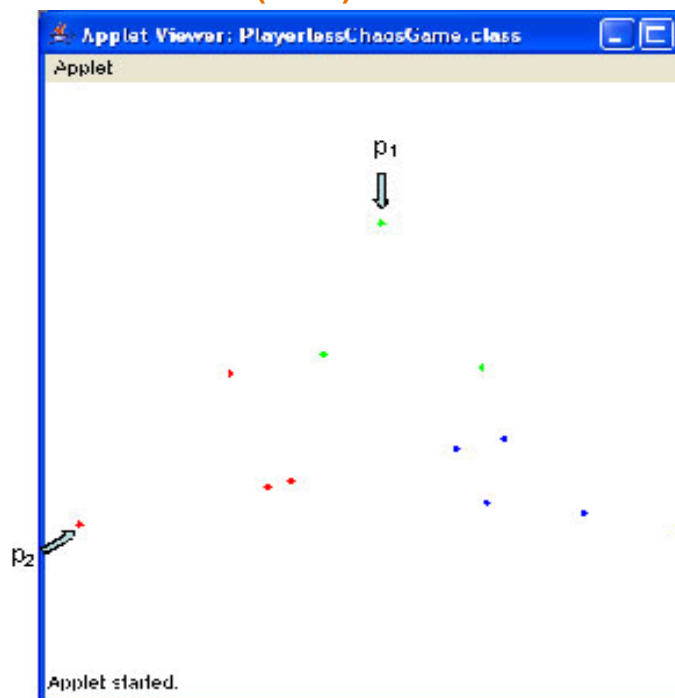


Figure 1. First Few Iterations of the Drawing Algorithm

### Tips, Hints, and Tricks: Consider the following:

To implement the three-sided die, take advantage of the `nextInt` method of the

`java.util.Random` class.

- An easy way to draw a single pixel is to draw a line for which the beginning and ending points are the same.
- Calculating coordinates is a challenge in this activity. You may be tempted to use a `java.awt.Point` object to keep track of the coordinates of each point drawn. The problem is that `Point` objects store their coordinates as `int` values, which can lead to rounding errors the more iterations you run. A better solution is to use objects from the `Point2D.Double` class from the `java.awt.geom` package. These objects store their coordinates as `double` values. Note that you will need to convert the coordinates to `int` when drawing a point.

**Cross-Curricular Connections – Discussion:** Work with a partner or small group to answer these questions and discuss your results with the class:

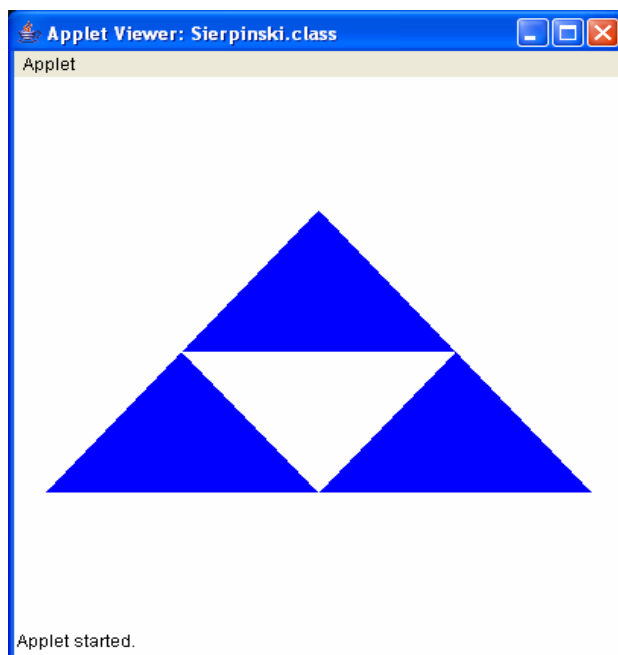
- Where did the concept of chaos originate?
- Who were the early mathematicians who laid out the principles of chaos?
- From what elements in nature did they derive their formulations?
- Some say that chaos theory is derived from observing of the universe and the endless transitions among celestial bodies. What are your thoughts on this?

## Activity 2: The Sierpinski Triangle

**Overview:** In this second activity, students will draw the same fractal using a completely different algorithm. This shape is called the Sierpinski triangle, named for the Polish mathematician Waclaw Sierpinski, who described the triangle and its properties in 1915. (The shape had already appeared in Italian art as early as the 13th century.) A good resource regarding Sierpinski and this triangle is <http://mathworld.wolfram.com/SierpinskiSieve.html>.

**Purpose:** To practice recursive programming, implementing an alternative algorithm that will converge on the same drawing as in Activity 1.

**Assessment:** How well do students design and implement the algorithm? How well do the students research and write answers to cross-curricular questions? See the attached rubric for details.



**Figure 2. Example of the Middle Triangle**

Given a triangle  $T$ , the middle triangle of  $T$  is formed by connecting the midpoint of every side of  $T$ . See Figure 2. The Sierpinski triangle is a drawing bounded by an equilateral (or any other) triangle. The interior of the triangle is divided into triangular regions that are either filled or unfilled. The Sierpinski triangle can be defined as follows:

- The level 0 Sierpinski triangle is simply a filled equilateral triangle.
- The level  $k$  Sierpinski triangle is found by going to every filled region of the level  $k-1$  Sierpinski triangle and unfilling the middle triangle. See Figure 1. This is a level 1 triangle. It is generated by going to the level 0 triangle and unfilling the “middle triangle”. This is the white portion in the middle of the triangle in Figure 1. To generate the level 2 triangle, consider the four triangles of level 1. For each of the filled triangles as represented in the level 1 triangle, the middle region will be unfilled.

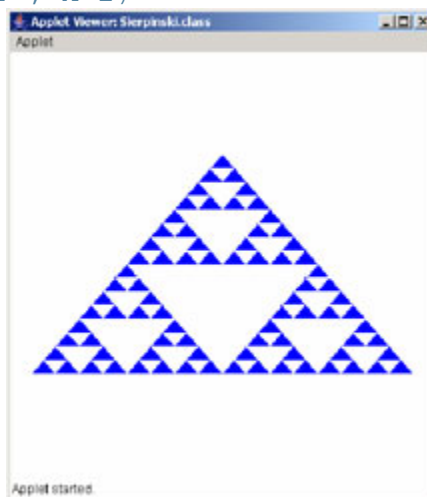
This leads to the following recursive algorithm used to draw the Sierpinski triangle:

```
Algorithm: DrawSierpinski(T, k)
Input: A filled triangle T and an integer k
Output: k Sierpinski levels drawn on T
```

```

If (k>0)
  Clear the middle of T
  For each of the three remaining filled triangles T' inside T
    DrawSierpinski(T', k-1)

```



**Figure 3. Level 4 Sierpinski Triangle**

This activity consists of using this algorithm to build an applet to draw the Sierpinski triangle. Experiment with different levels of recursion.

**Tips, Hints, and Tricks:** Consider the following:

The best way to clear the middle triangle is to draw in `XOR` mode, alternating between two colors when drawing pixels. That is, if the drawing color is red and the `XOR` color is blue, then use red the first time you draw a pixel, and the next time use blue. To learn more, see the `setXORMode` and `setPaintMode` methods of the `java.awt.Graphics` class.

As in Activity 1, the `Point2D.Double` class is important here.

One interesting extension is to see if you can modify your code so that you can draw the three major sub-triangles with the same colors as in Activity 1.

**Writing and Reflection Prompts:** Work with a partner or small group to answer these questions:

- Describe Sierpinski's work in developing his famed triangle. What do you think was the inspiration for his work?
- How does your program implement the Sierpinski triangle?
- What algorithms did you use and how do they differ from the algorithms you implemented in the first chaos assignment?

Once you have arrived at the answers above, write a short, reflective paper (three to five paragraphs) detailing your individual and collective discoveries in Activities 1 and 2. Alternately, you can write about a specific aspect of chaos theory, Sierpinski's work and life, or fractals.

### Activity 3: The Chaos Game

**Overview:** This final activity will combine what students have done in the previous two activities. Designed for one player, the Chaos Game is to be played on a Sierpinski triangle of a level between 2 and 6 (2 is easy, 6 is very hard). The game will begin with the player's "game piece" (a point) on vertex p1 and a randomly selected lowest level, with filled triangles highlighted.

The aim of the Chaos Game is for players to move their game piece from the start position into the highlighted triangle in as few moves as possible. Moves in the game are the exact same moves made in Activity 1:

The player selects one of the three original vertices and moves the game piece from its current position to the point halfway between its current position and the selected vertex.

**Purpose:** To design and implement a complicated application.

**Assessment:** How well do students design and implement the game? How well do they research and write answers to cross-curricular questions? See the attached rubric for details.

To play this game, write an applet for a particular level of difficulty. The applet should show the number of moves and possibly the statistics for each player. The design of the user interface for the game is up to you. Think about how the game is played and come up with a convenient player interface.

**Tips, Hints, and Tricks:** The big challenge of this activity is to design a user interface for the game that makes it easy and intuitive for a player to play. You need to handle different player contingencies, such as the following:

- Starting a new game
- Restarting a game
- Selecting a difficulty (recursion) level
- Making a move and not winning
- Making a move and winning

Use your knowledge of Swing to allow a player to perform each task and get useful feedback. For example, you might decide to use small buttons at each vertex of the Sierpinski triangle to represent the three possible moves at any position. You might also use a small circle to represent the game piece. Then, when users make a move by clicking a button, they will get feedback by seeing the new position of the game piece.

**Writing and Reflective Prompts:** This game is a logical extension of the first chaos assignment. Answer the following questions in a small group or with a partner:

- How did the first assignment help you to get started on this final project?
- What modifications did you have to make to that assignment to get the final assignment to work?
- What new algorithm did you need to develop and how did you implement that?
- How would you use what you have learned here to model theories about movements within the universe—maybe even the big bang theory?

**> Pacing/Timeline**

- Activity 1: 6 hours
- Activity 2: 6 hours
- Activity 3: 12–24 hours

**> Teacher Reflection (For example, what worked well in this project? What would you change if you were to teach it again?)**

- Were students engaged in the gaming activity at the expense of making connections to other cross-curricular activities and discussions?
- Did students develop more than a surface curiosity for the activities?
- What teaching strategies could be used to enhance the activities and classroom discussions?
- Are there opportunities to work collaboratively with other educators in science, math, or technology on this project? If so, how would this help or hinder student comprehension and skill development?

## > The Chaos Game - Rubric

Attribute	Needs Improvement	Proficient	Advanced	Maximum Pts.
Creates Die Class	Class not well defined – lacking constructor(s), accessor(s)	Class defined with constructor(s), accessor methods	Class fully defined with constructor(s), action(s), accessors(s)	
<i>Creates Game Playing class</i>	Class not well defined – lacking constructor(s), accessor(s)	Class defined with constructor(s), paint and play methods=	Class fully defined with onstructor(s), paint and play methods, methods to establish vertices and generate new points	

### Algorithm Development

<i>Rolls Die</i>	Fails to define method	Generates and returns a random value	Generates and returns a random value
<i>Establishes Anchors</i>	Fails to define method	Establishes anchor points that define an equilateral triangle	Color codes equilateral triangle as specified; triangle occupies the better portion of the drawing surface
<i>Generates Points</i>	Fails to define method	Correctly establishes coordinates for new point	Correctly establishes coordinates and sets colors
<i>Screen Painting</i>	Fails to paint triangle	Paints once or paints all at once	Appropriately paints and repaints with wait time between calls

## Writing and Reflection

<i>Grammar and Style</i>	Makes some grammatical, punctuation, spelling, or usage errors	Makes a few grammatical punctuation, or spelling errors, although most do not interfere with the overall message or the substance of the text	Writes well. Makes no obvious grammatical or punctuation errors. A pleasure to read.
<i>Content</i>	Presents little information about the history of chaos theory or the Sierpinski triangle	Presents some information and facts showing a basic understanding of the principles discussed and their importance	Presents many interesting facts and information that reveal an interest in and understanding of the principles and concepts discussed
<i>Sources</i>	Presents little or no research materials. Sources may be dubious in nature.	Uses some substantive Internet sources and texts that appear to be reliable	Cites original materials and Internet sources in ways that demonstrate a thorough search and understanding of the topic(s)