

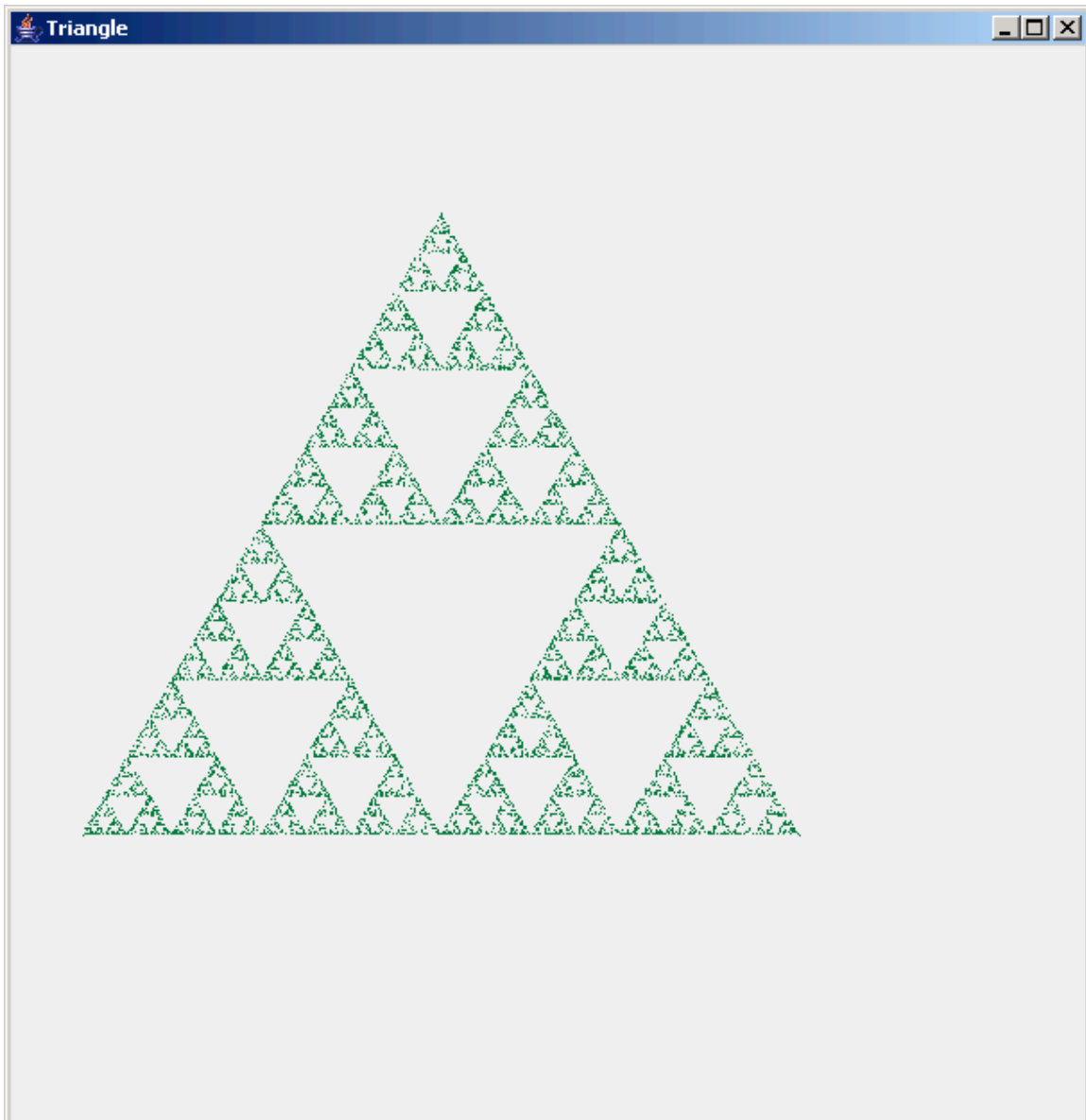
CS1322 Homework 4: Sierpinski's Gasket

(Due: Tuesday, Sep.27, 2005 @ 6:00 pm)

For this assignment, you'll be writing two classes that will prompt the user as follows:

```
cs\ C:\WINDOWS\system32\cmd.exe - java Triangle
C:\My Documents\CS1322\fall06\hw4\soln>java Triangle
Number of iterations:10000
```

And then display a window with the following graphic:



Requirements

You'll be writing two classes: Triangle and TrianglePanel. Class Triangle is similar to several examples you have seen in class and can find in the book. It creates a JFrame, calling all the usual initialization methods, and adds a TrianglePanel to the content pane. A good example from the book of this Class and ClassPanel relationship is on page 177-178: SmilingFace and SmilingFacePanel. The only additional thing you will need to have in Triangle, before any of the JFrame initialization, is a prompt for the user to input how many **iterations** they want the program to perform. The above picture is generated by 10,000 iterations (i.e., it contains 10,000 points).

Class TrianglePanel extends JPanel. You will need to write at least two methods for the TrianglePanel class: a constructor that takes in an int (the number of iterations, passed from Triangle) and the usual paintComponent method. You will also need to declare at least one field in TrianglePanel: an int that stores the number of iterations. You can use additional helper methods if you wish. Now let's talk about the drawing algorithm itself.

Drawing the gasket

Sierpinski's Gasket starts with three points: p1, p2, and p3. (These points are the vertices of the outermost triangle in the figure above). For this assignment, we will pick the following points: p1=(40, 440), p2=(240, 93.59), and p3=(440, 440). Your paintComponent method will first draw a dot at each of these three points and then pick one of them at random; we call this point the **current point, cp**. Then, the method will start a loop from 1 to the number of iterations specified by the user. For each iteration, the method will:

- Pick one of the three original points (p1, p2, or p3) at random
- Find the midpoint between the randomly chosen point and your current point **cp**
- Draw a dot at the midpoint; the midpoint becomes your current point **cp** for the next iteration.

That's it! If you implement the classes correctly, a drawing like the one in the figure above will magically appear!

Hints/Notes

- Feel free to use any reasonable background-foreground color scheme you want.
- Use a Random object to choose which of the three points you use for each iteration.
- There is no drawPoint method in class Graphics, but you can draw a dot by drawing a rectangle with height and width of 1
- Use two doubles to store each of your points—one for the x and one for the y coordinate. Cast these doubles to ints when you pass them to the fillRectangle method. (Using doubles improves precision and results in a nicer-looking gasket.)
- The midpoint formula (the only formula you need to solve this problem) is:
$$x_m = (x_1 + x_2) / 2$$
$$y_m = (y_1 + y_2) / 2$$
- Solve this problem iteratively (you'll need a for loop)!
- A panel of size 500x500 works pretty nicely

Turn-in

Triangle.java and TrianglePanel.java