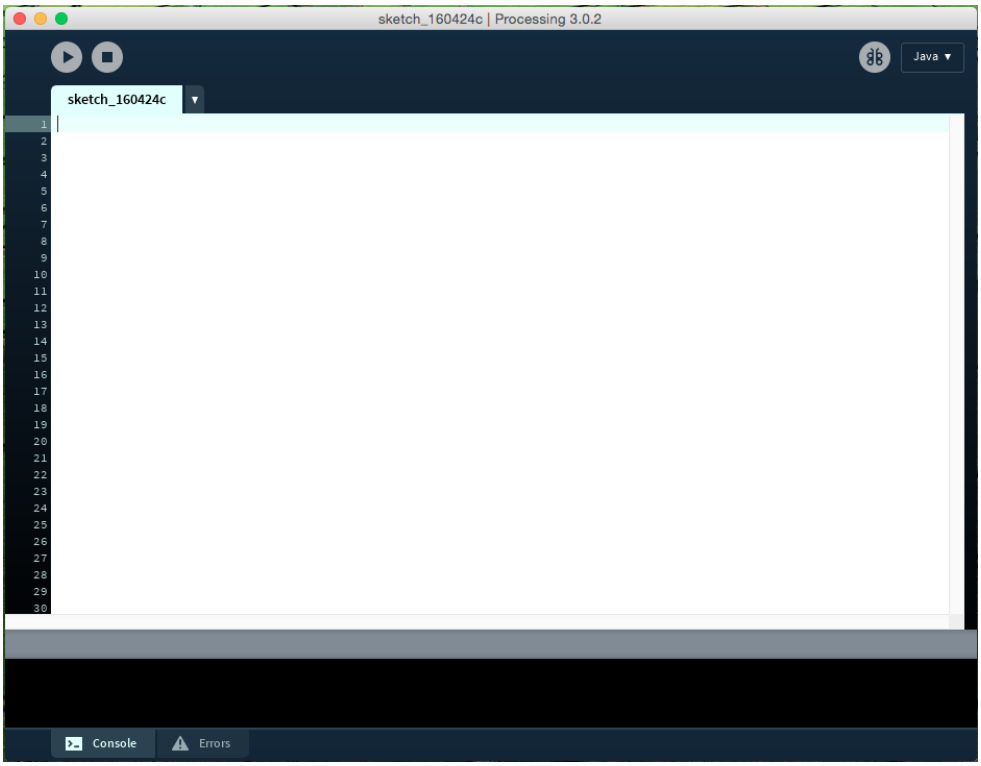
CS161: Introduction to Computer Science
Homework Assignment 11
Due: Wednesday 5/4 by 11:59pm

Now that you know the basics of Java, let's explore some of the more interesting, creative things that you can do! This assignment introduces you to *Processing* – a Java library for the visual arts.

If Java is like an industrial manufacturing plant, then Processing is like a maker space[1]. Processing uses a subset of Java to enable quick prototyping of ideas. Processing is both an *interactive development environment* or "IDE" for short – like BlueJ – and a collection of libraries that help you write visual programs called "sketches". Here's a picture of what the Processing IDE looks like:



In this assignment, you will download and familiarize yourself with Processing and then write a sketch that uses recursion to draw a Sierpinski Triangle.

[1] This analogy is courtesy of Prof. Mullen

An Introduction to Processing

1. Watch the "Hello Processing" tutorial on the Processing webpage: `https://processing.org/`
   To find this tutorial, click on "Tutorials" in the left sidebar. "Hello Processing" is the very first video option. This tutorial consists of 5 videos:

   - "Hello"
   - "Shapes"
   - "Color"
   - "Interact"
   - "Questions"
   - "Goodbye"

   and it should take about 1 hour to complete.

   The equivalent of Javadoc for Processing can be found by clicking on "Reference" in the lift sidebar. The reference section has a list of all functions that you can call along with an explanation of what input parameters they take.

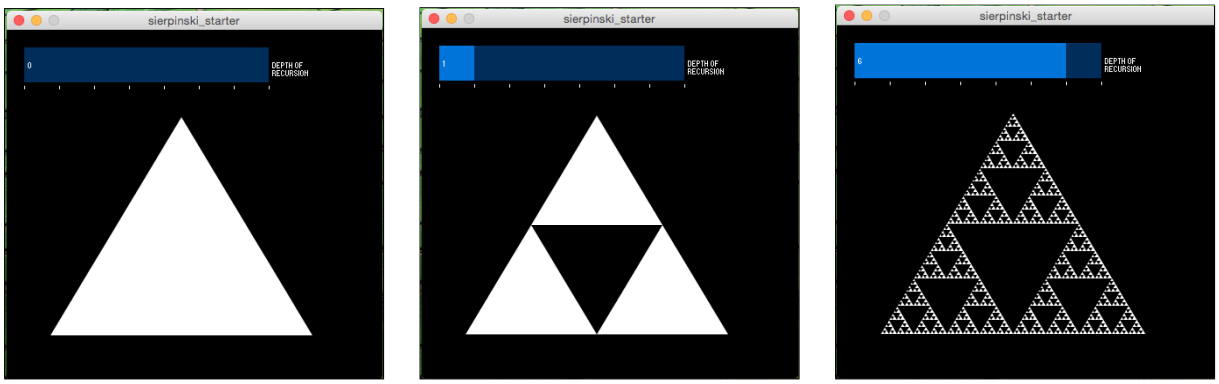
2. Download Processing by clicking on "Download" in the left sidebar. If you are using a lab machine, you can wait to download processing until lab. If you are using a laptop, please download Processing before coming to lab.

The Sierpinski Triangle

The Sierpinski Triangle is a fractal that can be drawn using the following algorithm:

1. Draw an equilateral triangle (i.e all sides have the same length)
2. Compute the midpoint of each side of the triangle
3. Remove the triangle defined by the midpoints. This leaves behind 3 smaller triangles: one on top, one on the bottom left, and one on the bottom right.
4. Repeat steps (2)-(3) recursively for each of the 3 smaller triangles.

For this assignment, you'll create a sketch that draws an interactive Sierpinski Triangle. Your finished assignment will look something like the image below, where the user can slide the value in the slider at the top of the screen to change the depth of the recursion (i.e. the slider controls the base case):



Download the starter code for the assignment. The slider has already been set up for you. There are two things you will need to do to complete this assignment:

1. The first is to implement the `midpoint()` method in the `Point` class. This method takes in another `Point` and computes the midpoint between *this* point and the *other* point. Conceptually, this is very similar to the `distance` method in the `Key` class.

   The midpoint between two points is simply the average of their x- and y- coordinates. For example, if I have a point at $(20, 100)$ and another point at $(50, 10)$, then the midpoint is:

$$x_{\text{mid}} = \frac{20 + 50}{2} = 35$$
$$y_{\text{mid}} = \frac{100 + 10}{2} = 55$$

   So, the midpoint is $(35, 55)$. It is sufficient to use integer division for this application.

2. The second is to implement the `sierpinski()` method. This is a recursive method. There is no way to "remove" a triangle as the algorithm above requires. Instead, you can draw a triangle (whose vertices are the midpoints) that matches the background color which will give the illusion that you have removed a piece of the triangle.

Feel free to play around with your code: change the colors, re-order the lines of code inside of the `sierpinski()` method, draw different shapes, etc. With recursion, it's always interesting to see what sorts of weird patterns you can produce.

Submitting your homework assignment

You should submit your `sierpinski_starter` folder, renamed, and zipped. Upload to Moodle.