Warm-Up Project: Linear Fractals
CS 3451, Spring 2017

The purpose of this project is to calibrate yourself with regard to the difficulty of the programming assignments of this course. You should start on this project before Friday! In addition, this project will familiarize you with the Processing.py programming environment, which we will use for our later projects. This project is due on Tuesday, Jan 17 at 11:55pm. Each day late after Tuesday will cause your grade on the project to drop by 5%, and we will not accept the project after four days beyond the due date.

Authorship Rules

Each student must work on this assignment on their own. You may not use code that anyone other than yourself has written. The only exception to this is that you may use the example processing code that was provided by the instructor. Code that is explicitly not allowed includes code taken from the Web, from books, from other students, or from any other source other than yourself. If you are unable to complete this assignment on your own, you do not have the programming skills needed to successfully complete CS 3451.

Project Description

You will create an interactive program that will display a linear fractal. Your program will use the position of the mouse to set the value of a complex number of the form v = (a + bi). Your program will then draw a large collection of points (or small circles) that are positioned at the locations given by sums of powers of this value v. In particular, you will calculate truncated sums of an infinite series of powers of v:

0
$0 \pm v^0$
$0 \pm v^0 \pm v^1$
$0 \pm v^0 \pm v^1 \pm v^2$
$0 \pm v^0 \pm v^1 \pm v^2 \pm v^3$

The first term of this series represents the position (0,0), which should appear as a point in the center of the screen. The next term of the series is $0 \pm v^0$, which is the same as the two positions (1,0) and (-1,0). The next term gives four possible values, and you should draw a separate point for each of these values. The next term gives eight points, then 16, 32, 64, and so on. Your program should calculate at least the first 10 terms of this series, and plot points for each of these, which corresponds to drawing at least 1023 points.

Calculate powers of v using the standard rules for multiplying complex numbers.
For example $v^2 = (a + bi)(a + bi) = (a^2 - b^2) + 2abi$, where i = sqrt(-1).

Additional Requirements

To add more variety to the images that you draw, add some kind of color variation to your points (small circles). You may choose colors based on the positions of the points, based on the truncation depth for the point, based on whether you are on a positive or negative branch of the series, or anything other scheme of your own choosing.

The region of the complex plane that you display should encompass at least values in the range of -3 to 3 for both the real and imaginary axes.

You should continuously read the mouse location and draw a new figure based on the mouse position. Using the mouse location in the window, your program should set the real and complex components of v. Both components of v should be allowed to be either positive or negative. The easiest way to achieve this is for the center of the screen to be considered the position (0,0) for purposes of setting v. The x-coordinate

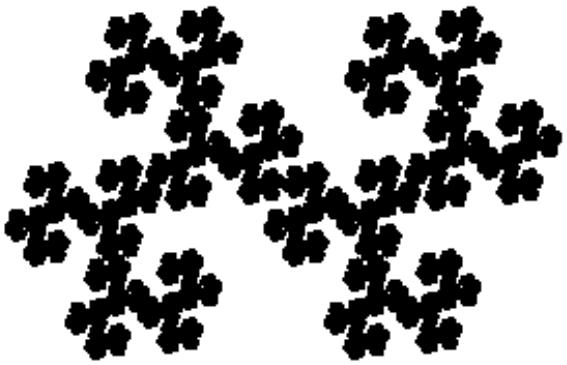
of the mouse should control the real portion of v, and the y-coordinate should set the imaginary component. Your program should allow the mouse to give real and complex value of v to be at least in the range of -2 to 2.

Your fractal program must be written in Processing.py, the python form of Processing. The Processing.py web site is here: http://py.processing.org. This web site includes downloadable versions of Processing for Windows, OS X, and Linux machines. It also has extensive documentation and examples. Processing.py is based on Python, and so it has all of the familiar Python language constructs.

Examples

Below are some examples of the forms that your program should be able to create. Keep in mind that your own program should add some color variation to the points.

v = (0.312, 0.576) :



v = (0.664, 0.336) :