Criterion B: Design

The program will be designed in three parts, the back end of classes and methods, the main JFrame where the graph is generated, and the GUI JFrame where the user would enter data.

Figure 1: UML diagram showing relationships of the back-end classes

The main StatsSaver class would contain the majority of global variables, inner classes and methods used generate the fractal to the panel.

The viewer/main GUI contains the swing generated interface for the user to initialize and manipulate the fractal that they choose to generate.

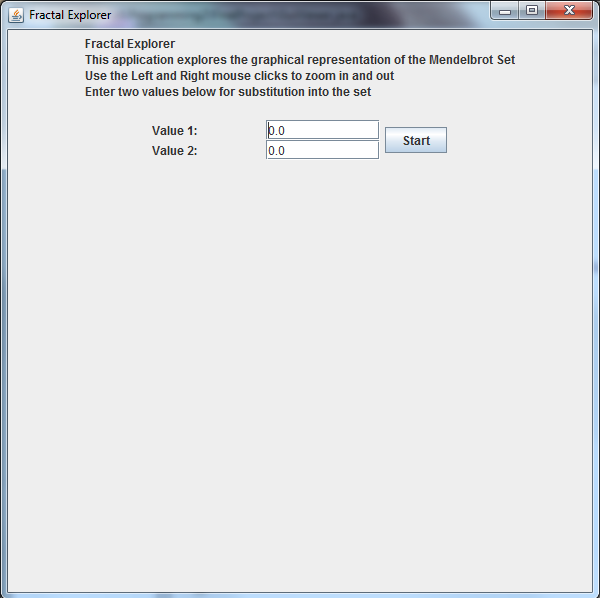
Each class (GUI viewer, frame, and main class) were tested separately with checkpoints and a text-based interface.

Figure 2: Screenshot of Initial User Interface

StatsMain.java is the entry point and a JFrame. It displays a basic JFrame with an instructionPanel, JTextField’s, and a “Start” Button.

The GUI contains user limitations on the text fields as well as action and key listeners for user events of clicking the “Start” button or pressing enter in the textfields.

The frame is also not resizable (there isn’t a reason for the user to rescale this simple GUI) and when the initializes a fractal they create a new JFrame.



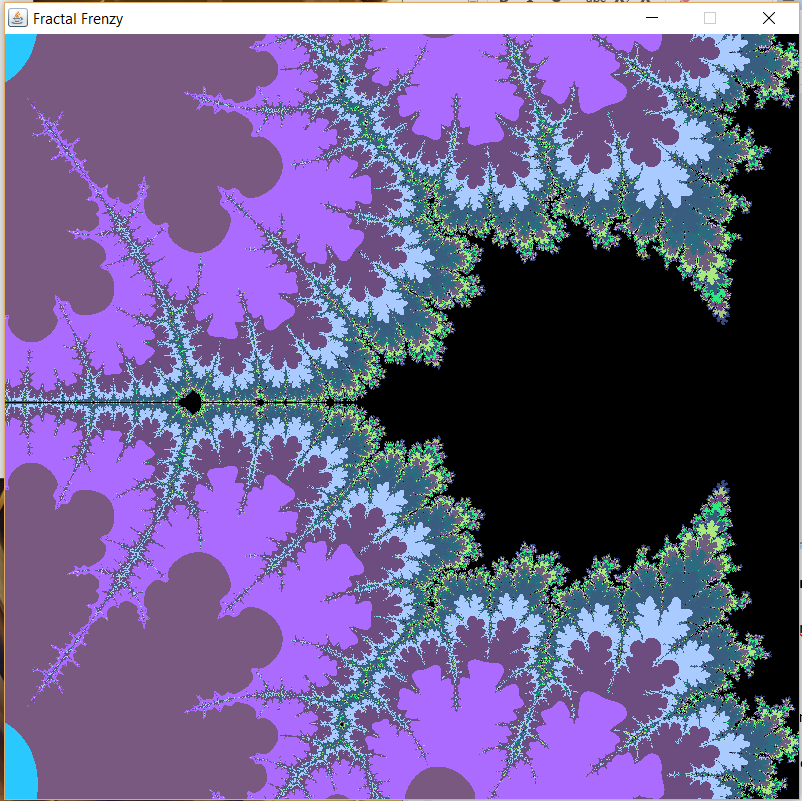


Figure 3: Fractal Image example drawn onto the JFrame

Fractal BufferedImage is drawn to screen using computeValues(), generateColor(), and update() methods that tests every single pixel in the Mandelbrot set through 200 iterations to determine compatibility then generates image.

User is allowed to manipulate image through left and right clicks of the mouse to zoom in and out as well as the W, A, S, D keys to move in the linear directions of up, left, down, and right, respectively.