This week we are going to get a bit arty. Fractals are a wonderful example of recursion-in-motion, of which there are some simple examples (Course Textbook chapter 3.6) and much more complicated examples that challenge how we think about the world potentially (the Mandelbrot Set for example). They also classically appear in art and have mathematic underpinnings that are really interesting.

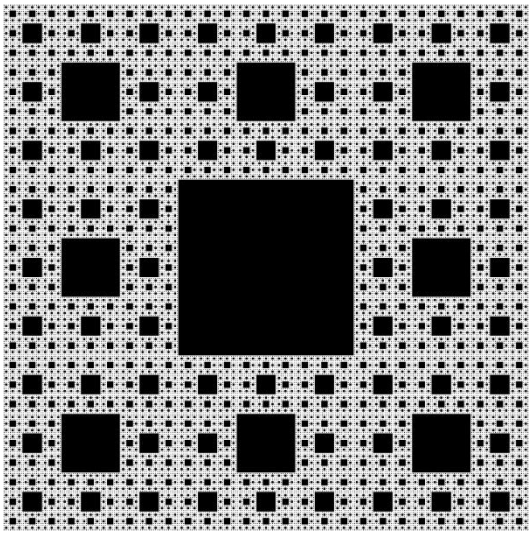
The Problem

You are going to use your knowledge of recursion, fractals, and JavaFX to make an interactive fractal art program. We are not going to get as in-depth as representing a dynamic Mandelbrot fractal, but we want to try and be artsy all the same (and the best way to be artsy is to be curious and open to experimenting!)

Your goal is to write a program in JavaFX that will display an empty canvas that the user can click on to cause a fractal to be drawn centered around where they clicked on the canvas. The user should be able to click on multiple spots and have the fractals overlap in an interesting way (color merging, transparency, something more interesting than flatly overriding the old color).

The Fractal

An important component of the fractal goes hand-in-hand with how we have been learning about recursion: the base case. While these can theoretically go on infinitely, you should consider (for whatever shape(s) you choose to draw) how many iterations of that shape is interesting and **how they should change** at each 'level'. Consider the Sierpinski carpet:



Borrowed from the wikipedia page for Fractals

While the basic shape is a square, the size and placement vary in interesting ways, and while it implies infinite depth of smaller squares there is clearly a cut-off point eventually. We can also

see the pattern of the shape's spreading in a 3x3 block around each larger square. Much more colorful and complex examples exist, but this shows the basic idea of what could be drawn when the user clicks on the program's canvas.

The Canvas

Now, the reason I have been using the term 'canvas' is because that's exactly what we're going to use! This page discusses the Canvas class in the JavaFX library, which works in concert with the Graphics Context class to give you a good tool for drawing in a way that feels more natural to this kind of task (figuring out how to use Rectangle objects and HBoxes to do this? No thanks)

You should experiment with these new classes to get a feel for how they work, specifically looking at the 'stroke' and 'fill' methods in the API and the 'setGlobalBlendMode'. The goal with this activity is to be creative and make something interesting, so try some stuff out and see if you can make weird things happen!

Really, taking thirty minutes to mess around with this in practical terms will teach you a lot about how this works, more than just reading the API and going straight to your final program would. Consult the API and try to apply what you see!

The Program

Once you're comfortable with your Canvas and your 'paint brushes', and once you've had some ideas for your fractal designs given the tools at your disposal, we can start thinking about our program! You will **not** need to provide a UML diagram or design justification for this activity, but I still expect to see your growing competency in program design in how you write this program.

Make a plan for your program for your own benefit: what kinds of recursive patterns do you want to draw? Where can you implement the logic of those patterns in your program given the tools you will be using (Canvas, GraphicsContext, JavaFX in general)? How will the user interact with your program, and how will you handle overlapping fractals?

Technical Requirements

For the sake of variety, I expect to see **at least two** distinct recursive methods that draw different fractal patterns (note that by 'different' I am referring to the **shape/pattern** of the fractal, color variation is encouraged but not sufficient to differentiate two fractal patterns). Make sure you clearly label these methods as being your recursive drawing methods to help us easily identify that you have satisfied the requirements of this activity.

You will be assessed based on the quality of your code (comments, structure, readability, etc.), the quality of your recursive fractal drawing methods (no stack overflow incidents, successfully draws fractals, is actually recursive, at least two), and the quality of your program's user interface (how well you satisfy the above description of the problem).

Feel free to individualize your program's UI if there are particular features that you would like to include (changing the kind of fractal, customizing the color/fractal depth, etc.) to make the program feel more complete to you; while not required, that will help satisfy the requirement for program UI quality (and personalize your work!)

What to Submit

- Your code that implements the program described in this handout
    - Should be well commented (make sure you describe and justify each class)
    - Should be well structured (use of local methods, introduction of new classes, etc.)
    - Should specifically comment and explain the methods that draw your fractals (tell us your intent with each of them and any interesting context you want to provide)

Collaboration

While your submission will be solo for this activity, you are heavily encouraged to collaborate with your fellow classmates/peers when experimenting with Canvas/GraphicsContext to figure out what kinds of things you can do with those tools. Make sure that when you begin work on this activity, though, you are only presenting your own work.

The line between a general concept that can be discussed or worked through together and a specific challenge that you should complete solo can sometimes seem vague. The general rule is that in class I introduce general concepts to you that you need to understand, and in labs/activities I give you specific use-cases where the challenge is figuring out how to apply the general concept to the specific problem.

Helping one another with the general concepts is an admirable thing to do and is encouraged, but when it is time to test your abilities with an activity/lab/quiz your work must be your own.

An example would be that you can ask your classmates for help with how to write a recursive method properly, but you can't ask for help in how to write a recursive method that draws a fractal; the challenge of this activity is how we apply the general concept to this specific use case.

If you are ever uncertain about what you can collaborate with your fellow students on, please feel free to reach out and ask me so that I can clarify what is allowed with regards to these activities.

CS-2611 F23 ICA 6

Due: November 4’th, 2023 @ 9 PM

Late submissions are not accepted without prior discussion to extend the deadline.

I will not answer questions about the assignment asked after November 3’rd @ 9 PM.

This activity is expected to be completed solo. You may always discuss course concepts with your fellow students and how to approach a problem, but your work should be your own and only reflect your own ideas. Include in your comments any websites (such as Stack Overflow) that you consulted during your work and what contributions they made.

Information gathering is a valuable skill and knowing how to find information online is important to be able to do; letting me know what information you needed to seek out allows me to know what to focus on teaching in the future.