Trey Tuscai

Professor Al Madi

CS231 - A

27 September 2021

Project 2 Report

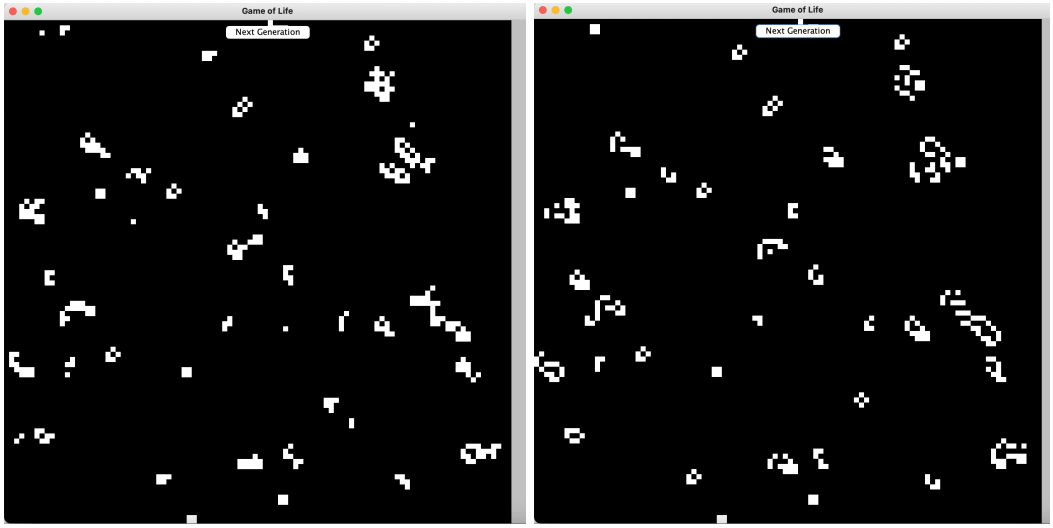
Abstract:

In this project, I created a version of Conway's Game of Life, which demonstrates the evolution of two types of cells, alive and dead. Both of them have specific rules that they follow like the alive cell stays alive if it has two or three living neighbors and the dead cell stays dead unless it has three living neighbors. Using a 2D array, I built a randomized grid of these cells, so each generation advances and updates its state according to the rules and number of iterations. Then, I implemented the Graphics class and the Swing package to develop a window based display. The Graphics class draws the alive cells as white rectangles and dead cells as black rectangles with the Swing package showing it on a grey JPanel in a JFrame window, so the evolution can be easily visualized. The purpose of this project was to develop concise code while familiarizing ourselves with 2D arrays, Java Graphics, and the Swing package.

Results:

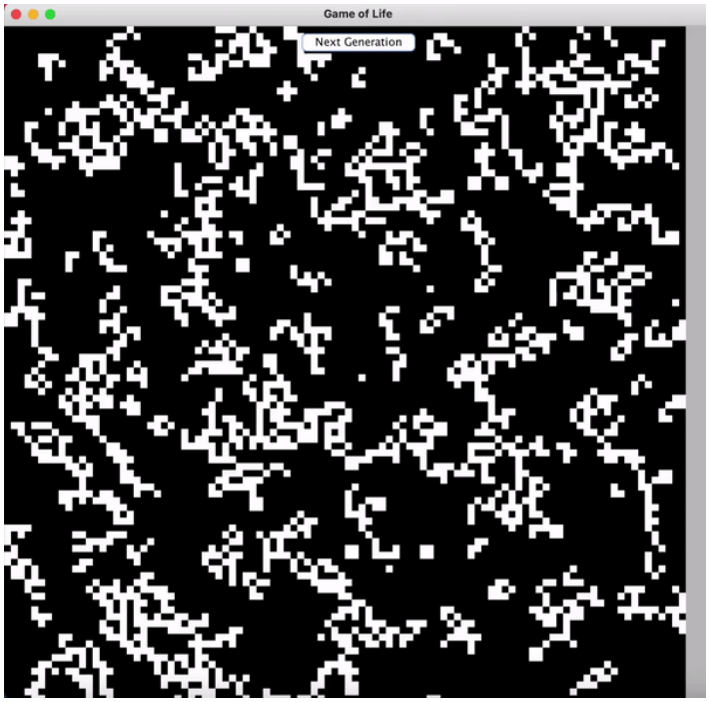
For my results, I ran the Game of Life four times to see the evolution of the cells. In the first game, I ran it using two generations of 10% of randomly positioned live cells. In games two through four, each game used 10 iterations of generational advancement on a 10%, 20%, and 30% density of randomly positioned alive cells. These games make sense and illustrate all the

new interactions between cells depending on the original density percentage while following the same original rules.

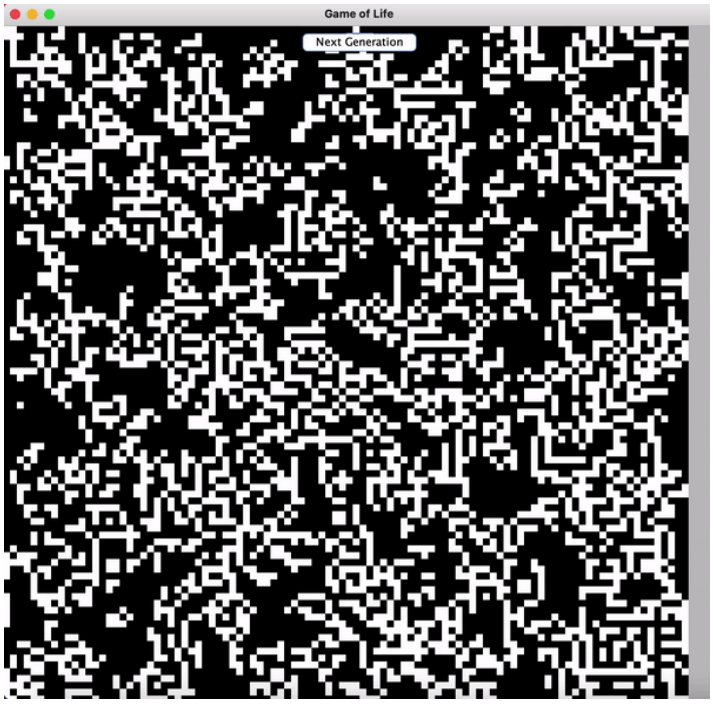


^ Output of first two generations of 10% alive cells

^ Output of 10% alive cells through 10 iterations



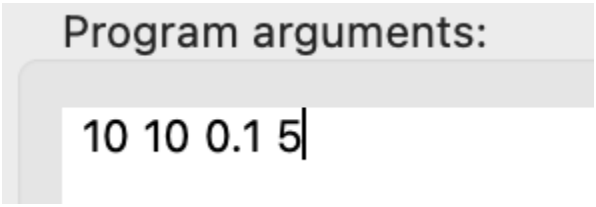
^ Output of 20% alive cells through 10 iterations

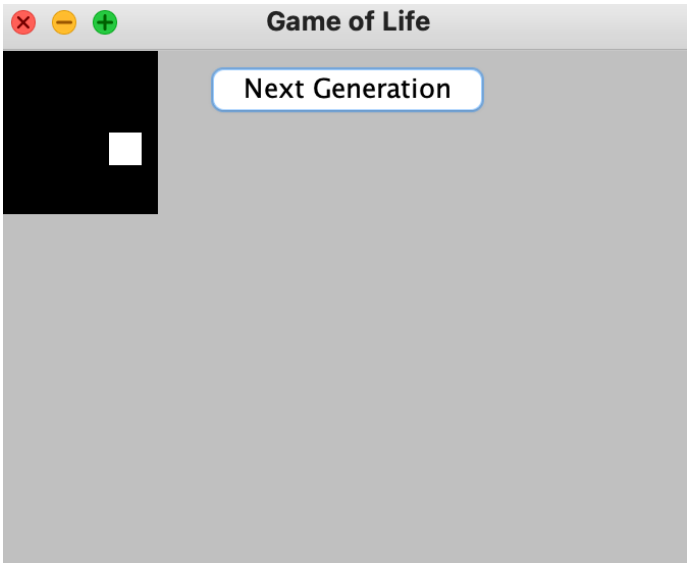


^ Output of 30% alive cells through 10 iterations

Extension: No. 1

For my first extension, I chose to make the game slightly interactable, which allows the user to choose the landscape height and width, density of alive cells at start, and number of update iterations. I used the Java Integer and Double class to parse command line arguments for the variables that controlled these parameters. scapeHeight was parsed to args[0], scapeWidth was parsed to args[1], density was parsed to args[2], and iterations were parsed to args[3]. When ints (scapeHeight, scapeWidth, iterations) and a double (density) are added to the command line, the user can fully control the aforementioned parameters of the game.

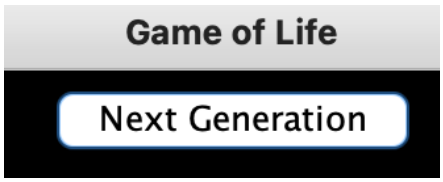




^ Example of arguments and the result ingame

Extension: No. 2

For my second extension, I chose to add a button that allows for extra iterations of evolution. On the JFrame, I added a JButton, which says “Next Generation.” Each time the button is pressed, the actionListener is notified. This queues one more iteration of cellular evolution in the game. I added this to allow the user to have more control over and to better see how each iteration works.



^ Example of JButton

References/Acknowledgments:

I received a lot of help with my Landscape and Cell class from Professor Al Madi. He assisted me with the advance, getNeighbor, and update state methods. I also used a few online resources.

https://docs.oracle.com/en/java

https://www.w3schools.com/java/

https://stackoverflow.com