Game of Life

Link to the versions: <https://github.com/0Plus0YiLong/Game-of-Life->

Testing Videos:

The Game of Life is a cellular automaton devised by the British mathematician John Horton Conway in 1970. User will initially input variables at the start but require no further input once the game starts.

Game obeys in 4 basic rules:

1. Any live cell with fewer than two live neighbours die.
2. Any live cell with two or three live neighbours lives on to the next generation.
3. Any live cell with more than three live neighbours dies.
4. Any dead cell with exactly three live neighbours

This project is created using Pygame, which is a cross-platform set of Python modules that consists computer graphics and sound libraries that can be used to create Graphical User Interface.

I have used these complexed techniques:

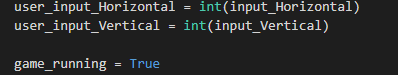
* A programmed GUI
* Object oriented programming using classes and objects defined by student.

**The program performs specific task:**

Our task is to make a unique version of Conway’s Game of Life. I have made a version that has a graphical user interface, a pause and start button that the user is able to pause between every generation, and the ability allow user to input integers to adjust their window size.

**Two or more variable types are used:**

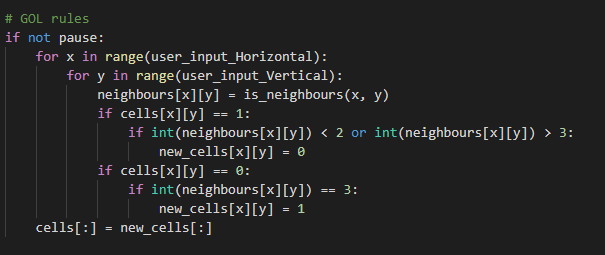
I have used integers for the customisation of the size of the window, as users can type integers and jusity the size of their map.



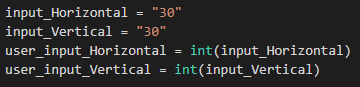
And I have used boolean variables to determine if the cell is alive or not, the game is running or it has been paused.

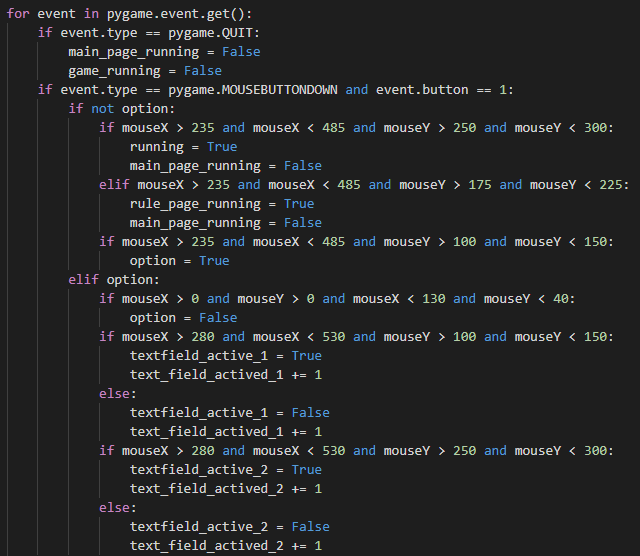
**Sequence, iterations and selection control structures are used:**

Sequence is present at when I used DrawRect to draw out the square shapes one next to eachother.

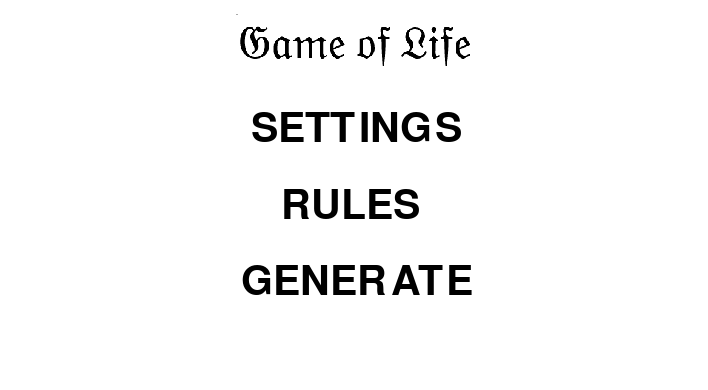
Iterative structure is also involved in the program, I have combined the selection structure with it, and use it to determine the number of neighbours of a live cell, and apply the rule of Game of Life and calculate the next generation.

**The program allows users to input typical data:**

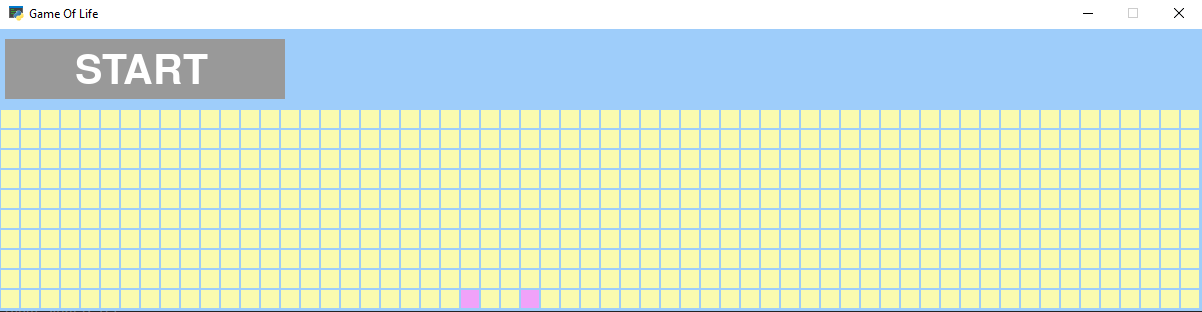
I have added the feature of allowing users to input typical datas like the width & length to their preference, make a cell alive by simply clicking a square.

This is shown here as I records the coordination of the mouse, whenenver the user clicks, and is detected to be within the range of a square, then that box will change its colour, which represents that it is a live cell. The user is also able to click on a live cell to make it a dead cell.

**At least two complex techniques have been used:**

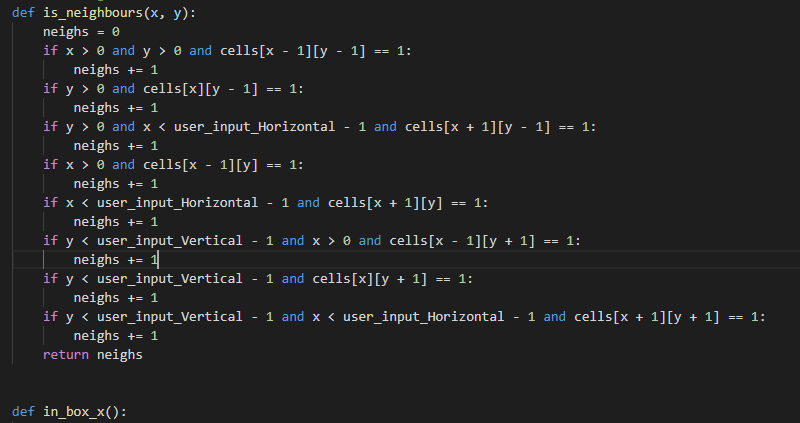
**GUI(Graphical User Interface)**

I have include the contruction of a graphical user interface within my project of GOF. My GUI is based on Pygame, by creating a GUI, I’m allowed to design a main page and better looking game of life. I can change the colour of the boxes, buttons.



**Class and Objects defined**

I have used a lot of functions that is defined by myself for different purposes, e.g. the calculation of number of neighbours. The whole project is done in the main class.



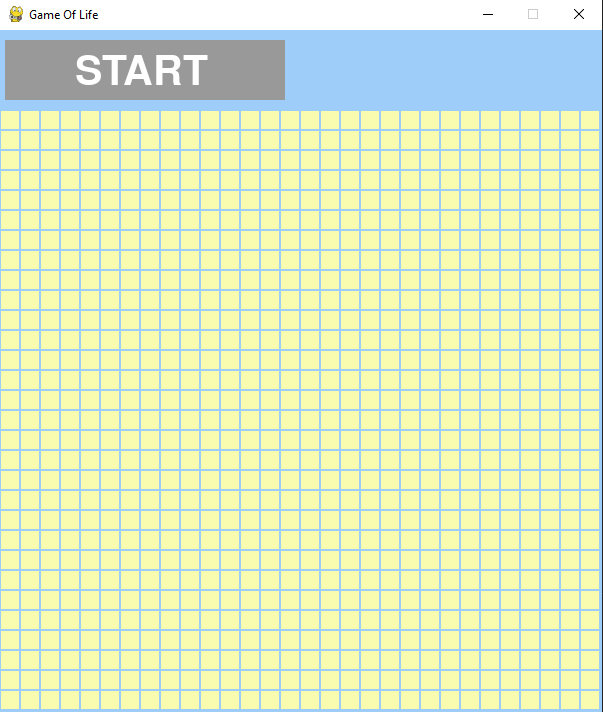
**Layout:**

I have also commented sections of my code, and is laid out neatly.  

**Version Control, Testings and Debug processes:**

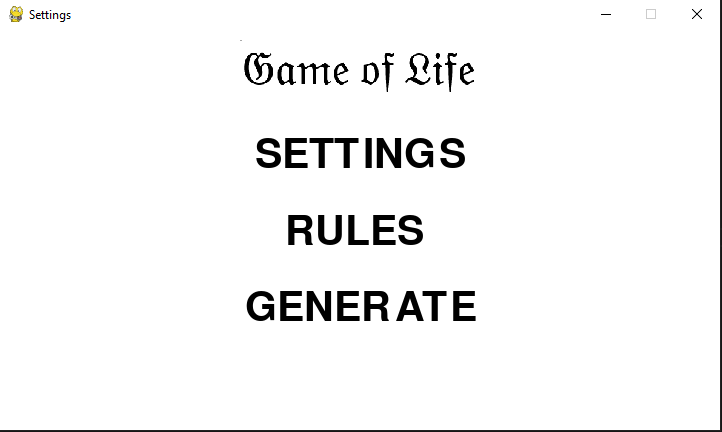
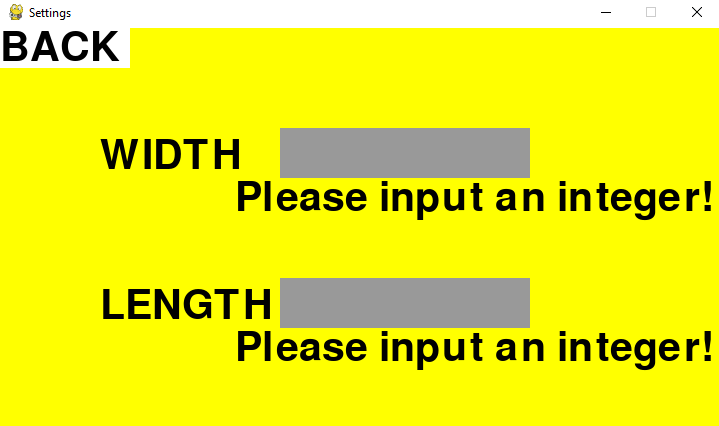
Along making the game, I have recorded versions where I have made big changes and bug fixes. I have done this to ensure that all components of my application work fine and as how I expected it to work. (All bugs are easier to be demonstrated by videos of testings)

Below are the 4 main versions including the bug fixes on each previous version.

**Version 1:**

I have created a basic GUI involving a start button, and a 30 x 30 grid “spreadsheet” using “Draw.rect”. With the basic rules of Game of Life implemented within. The user can use their mouse to click on the grids, once they do that, the box that they clicked will light up and become a “live cell”. I did this using “pygame.get.mouse.pos()” to catch the value of the mouses X axis and Y axis, to get the exact position of the mouse on the “spreadsheet”. Everything worked fine; however, it is limited to only the game itself, and has no other features e.g. User input variable, adjustment of the size.

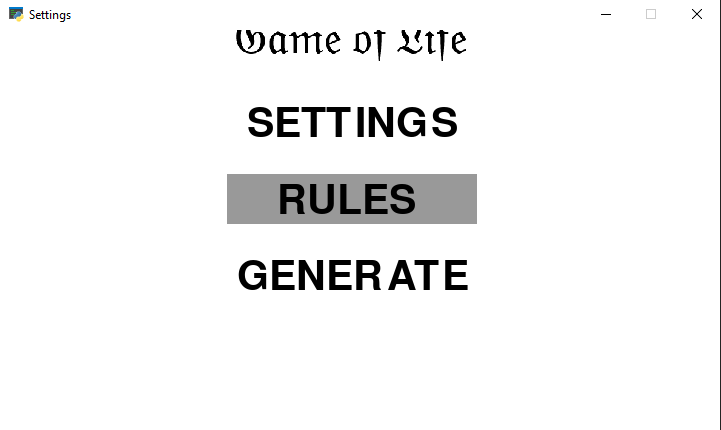
**Version 2:**

In the second version, I have added a main page that has three buttons, Settings, Rules and Generate. The settings button will take the user to a window, and they can change the size of the application by typing in the width and length. The Rules button will take the users to a window that displays the rules of Game of Life, how it works. The Generate button is basically to generate the size of the map of the width and length the user has inputted. If they didn’t input any number, the default size will be 30 x 30. However, the rules button does not work, and the “text input” codes have some logical problems, causing it unable to read what user has inputted, also it keeps validating the user input.

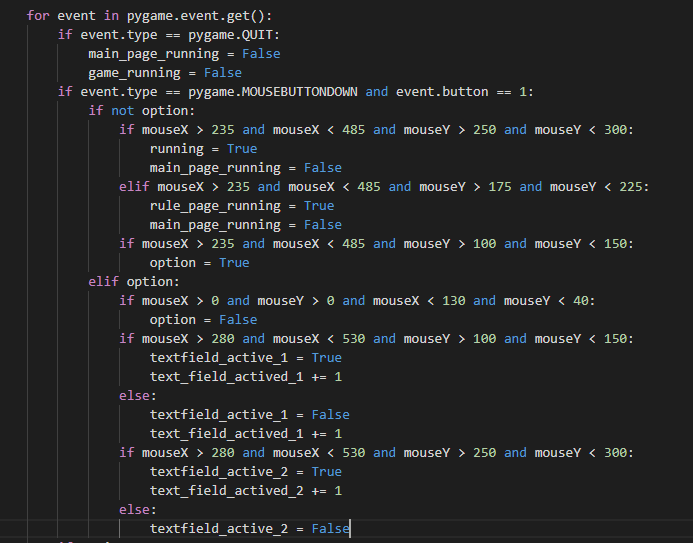
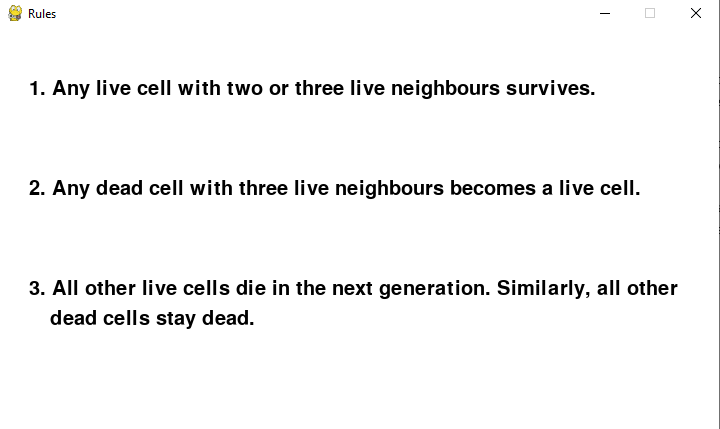
As shown on the left, it looks very untidy, and it uses a lot of memory that eventually makes the application to run at a lower speed.

And the rules button didn’t work.

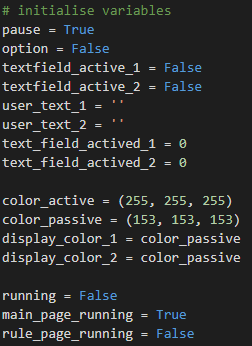
**Version 3:**

The third version, I improved the logic of my code. The problem of users unable to type anything into the text bar. Not much has changed on the design. I have disabled the rules window but kept the button in the main page. And the validation problem still unsolved. 

**Final version:**

In the final version, I managed to add back the Rules window, put in the basic rules in there, which I got from Wikipedia and expressed into simpler sentences. Before, when click Rules button, and the button works now. The non-stop validation problem is also solved, it was caused by an extra If statement that I didn’t found out earlier, and the map is able to be customized by the user’s input with no issue.

**The program is flexible and robust.**

I have made my program flexible and robust by using simple variables and control structures efficiently and different functions to simplify the codes throughout the development.

By validating the user input, whether it is appropriate or not. As when user is trying to input the size of the map, only integers are allowed. 