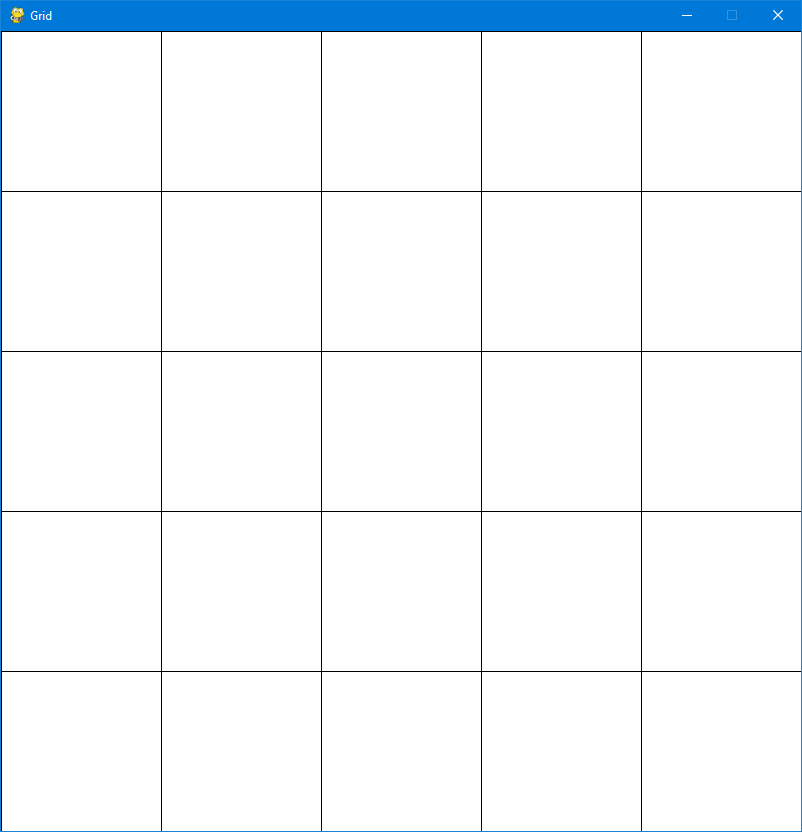
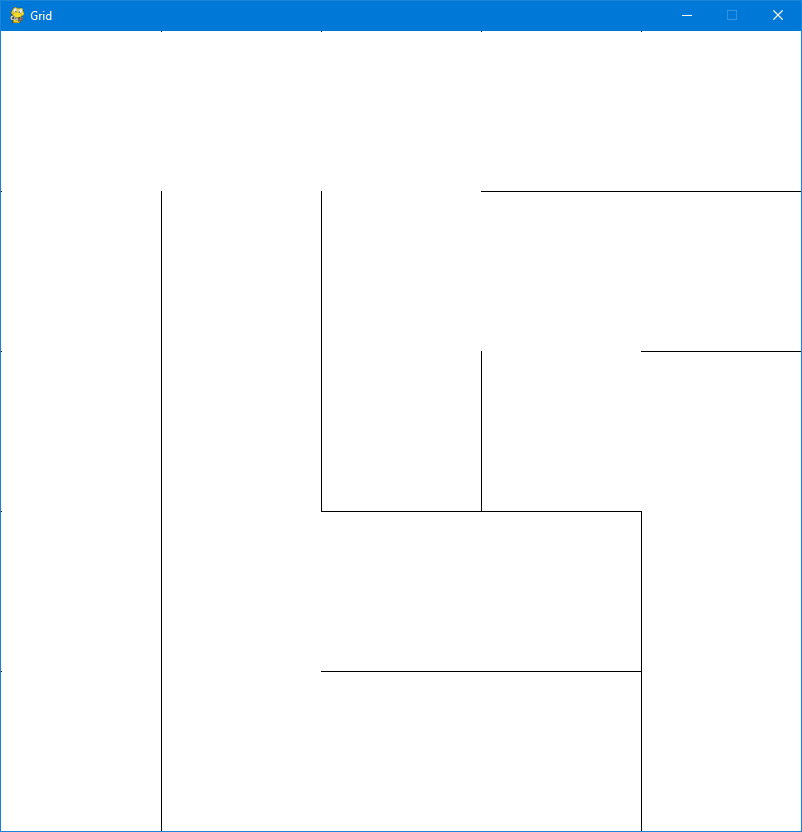
Creating the grid initially was somewhat difficult. I didn’t have any experience in Python so the formatting of Python had me a bit dazed, although I got somewhat of a grip after time passed. When I was thinking about it, I realised that the resolution of the screen was fixed to 800 by 800 pixels so it was already a square, and the maze had to have a n x n grid. Dividing 800 by the “n” value inputted would give equidistance across the screen “n” times. Knowing that, I found a way through using a “for loop” and the pygame.draw.line() function drawing lines starting at those n distances and going across the screen vertically and horizontally would result in a grid.



The grid had been created and now to transform it into a maze. Looking at the example of the binary tree algorithm, I noticed that they started at the top left of the grid and worked their way across and down the grid. While they carved a pathway, the carving would either only take the top part of the cell or the right part of the cell. It resembled a for loop interaction from my experience of C++ and Java. I created a maze by using a for loop again and drawing rectangles that would only either cut the top part of a cell or the right part of the cell. The pathing was determined by a random number generator that would have a 50/50 chance of cutting right or cutting above.

Example of a 5x5 maze



Example of a 20x20 maze

