Final Project Design Documentation

Background:

Hello! Welcome to the Design Documentation for my ENAE380 Final Project. My project was a simulation of how a plant's roots would grow through the ground.

The idea for this project was originally created around a year ago. Over the pandemic my mom started having knee issues and wasn't able to take care of her curry plants, these giant six foot mammoths that are older than I am. I started taking care of them in her stead, and found a love for gardening. During that time I found myself weeding the garden frequently, pulling invading plants out of the ground, and each time I was fascinated by the shape the roots had taken. Since then I had wondered if there was some way I could simulate this growth myself, as I couldn't find anything online that was similar. Thus, this project was born.

Technical Implementation:

The driving data structure for this project is a Binary Tree. A Binary Tree is a structure composed of nodes that point to children nodes.

- For my implementation, these nodes contained four pieces of data
    - X-Coordinate
    - Y-Coordinate
    - Pointer to the child on their left
    - Pointer to the child on their right
- If a node has no children, the left and right data pointers are set to None

I chose a Binary Tree for this project for a few reasons. At a given point, the data can branch and go both left and right. Through a recursive search method, it is very simple to reach the leaves of the tree.

To build an environment for the roots to exist in I created two 2-Dimensional Arrays:

- The first of these arrays is initialized with random values from 0-3.
    - 0 represents a stone or other physical barrier that roots cannot grow through.
    - 1-3 represent increasing levels of nutrient density.
    - I then created a dictionary that represents the numbers 0-3 with colors. 0 is given gray so as to look like a stone.

    - 1-3 are increasingly darker shades of brown
        - This was a stylistic decision since to my experience, dirt that was lighter in color was usually more dried and not as good for plants.
    - This dictionary and 2D array are then given to Tkinter, which draws it out and displays it.
        - This 2D array is also used to record the current location of the root system, with values of -1 meaning that a root has grown in that cell.
        - Roots are given the color white to make them distinct from anything else in the grid while still looking natural
- The second 2D array is a librarian that keeps track of where the root system has been. It is initially set to all zeroes as the root system has not been anywhere

After all the setup has been completed the main part of the code begins

- Using a recursive search algorithm, the program visits every leaf in the binary tree.
    - A leaf is any node that has no children.
- Upon getting to the leaf of a node, the program checks the cells surrounding it. Out of the eight surrounding cells, only five are checked
    - A stylistic choice that I made was not allowing roots to grow upwards. While this is certainly possible in real life, I felt that it would look nicer to have the roots only grow downwards.
    - As such,the five cells checked are:
        - Directly Left
        - Directly Right
        - Down and Left
        - Directly Below
        - Down and Right
- From these cells, a general location is established, and from there set of decisions is applied
    - The purpose of establishing location is to see if the roots have reached the borders of the grid. By checking where it is, any possible OutOfBounds Exceptions are averted.
    - A set of decisions is then applied
        - The root first looks at the left side and middle. From this it decides the optimal place to grow based on which cell has the highest nutrient density.
        - The roots consider growing diagonally last as this requires the addition of two cells. Rather than growing directly diagonally, the plant grows in a L shape to grow diagonally. Thus it must grow through an extra adjacent cell to reach the diagonal location

            - Once a destination has been decided, the librarian and map of the grid are updated with the new root location.
        - The same process is repeated with the right side, with straight down being an option if the left side hasn't grown there already.

    - There are two important restrictions that are always checked regardless of general location.
        - A root will never grow into a cell that is occupied by another root. This is to ensure that there is no overlap in the system.
        - The second restriction is another stylistic one that was recommended to me by a friend.
            - A root will not grow into a cell if there is a root adjacent to that cell.
            - This one is somewhat odd without context.
                - Given how small the grid is, a root takes up a significant amount of space.
                - Especially in smaller grids, if roots can grow adjacently to other roots, the path of growth quickly begins to look like an undefined blob rather than individual branching growth. This restriction forces the root system to expand through new territories.
                - This is another restriction that obviously does exist in nature. With the freedom of a larger grid or a third dimension, this restriction could be lifted.

All in all this project was a far greater success than I ever expected. The look of the roots growing through the grid is everything that I wanted out of this project. Looking back, there are a few things that I wish to improve on and would keep in mind for the future.

Firstly, TKinters inefficiency. As far as I could understand from Tkinter, the window had to be closed so that the new root system could be drawn. This process is slow and tedious due to the next issue that I could not figure out. This was the entire reason that screenshotter.py was created, so that no one would have to sit there and keep closing out the windows.
This section above was the biggest issue I had when making this project, but is a problem I have since fixed.

The drawn grid of Tkinter is laggy and resource intensive. Despite only using a simple system of cells to draw the grid, Tkinter struggled to draw a 70x70 grid, and could not open an 80x80 on my computer.

When making this project, I wanted to use X and Y coordinates to represent the location of roots on the board. However, in the 2D array representations of data, a cell is located by [row,col] and so for all of those, the x and y are placed backwards. This wasn't an issue so much as it was annoying to work with, and in the future I would probably just use row and col.

Finally, the separate librarian array is unnecessary. It could be combined with the main 2D array since I have roots represented there as well. I made it to keep things separate for myself, but it honestly isn't necessary.

This project was a lot of fun to make, and I'm certainly not done with it yet. I like the base that I've created, but in the future I would like to take this to 3D space, as well as a significantly larger grid. For now though, thank you for reading and have a nice day :)