Infinite Snail Board Extra Credit

Adam Hoffmeister

ECE 275

Running the Program and Viewing the Simulation

Run the main.py program in the file (preferably by opening the folder in an IDE with python installed). The program will first open the terminal, which will ask the user to either input 1, for a slow speed (5 fps), or 2 for a fast paced simulation (60 fps). After the user enters the preferred value, the simulation will open up in a separate tab. If there are any issues with running the code, please let me know.

Overview

The board will be covered in black squares and will not be revealed to be a colored square until the snail steps on that given square. The snail will begin on a colored square and continue it's movement from that point. The snail will be represented by a brown rectangle, smaller than the size of the other squares, and is always displayed in the center of the screen. The display of the board is set to a size of 7x7.

For simplicity, the simulation was designed to have a set of 10 colors. Each of the 10 colors are assigned their own delta-orientation, which changes the direction of the snail upon reaching that color.

Design

The program consists of a few objects: board, snail, cell, color circle, and node. The board class stores a dictionary which assigns the position of a given cell to the key, and assigns the cell object to the value. Whenever the snail uncovers a new cell, it is added to the dictionary. When the board is displayed, the board will check which uncovered cells are within the 7x7 display range of the screen. The board also initializes a snail object. The snail class stores simple information about the position and orientation of the snail, as well as a set of methods which move the snail and update its position.

The file labeled generate.py creates the colors and how each color changes. The program first generates a series of random colors and assigns a delta-orientation to each of those colors. Each of the colors are stored in a modified linked list, where each node holds two values, the delta-orientation and the color of the cell. Furthermore, unlike a traditional linked list, the tail points to the head. When each cell changes color, it simply moves to the next node within this modified linked list, allowing the given cell to continuously change color even if the cell is visited multiple times.

Future Optimizations

The primary changes are aesthetic designs. The snail is poorly represented by a brown rectangle and could be visually updated. Furthermore the display of the board should optionally be adjustable.