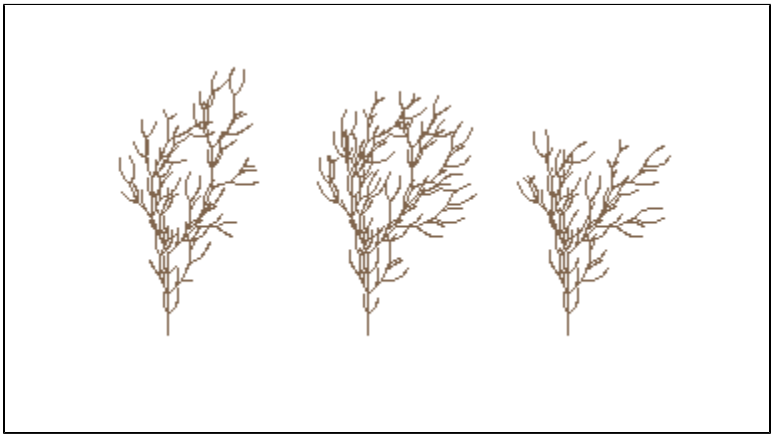
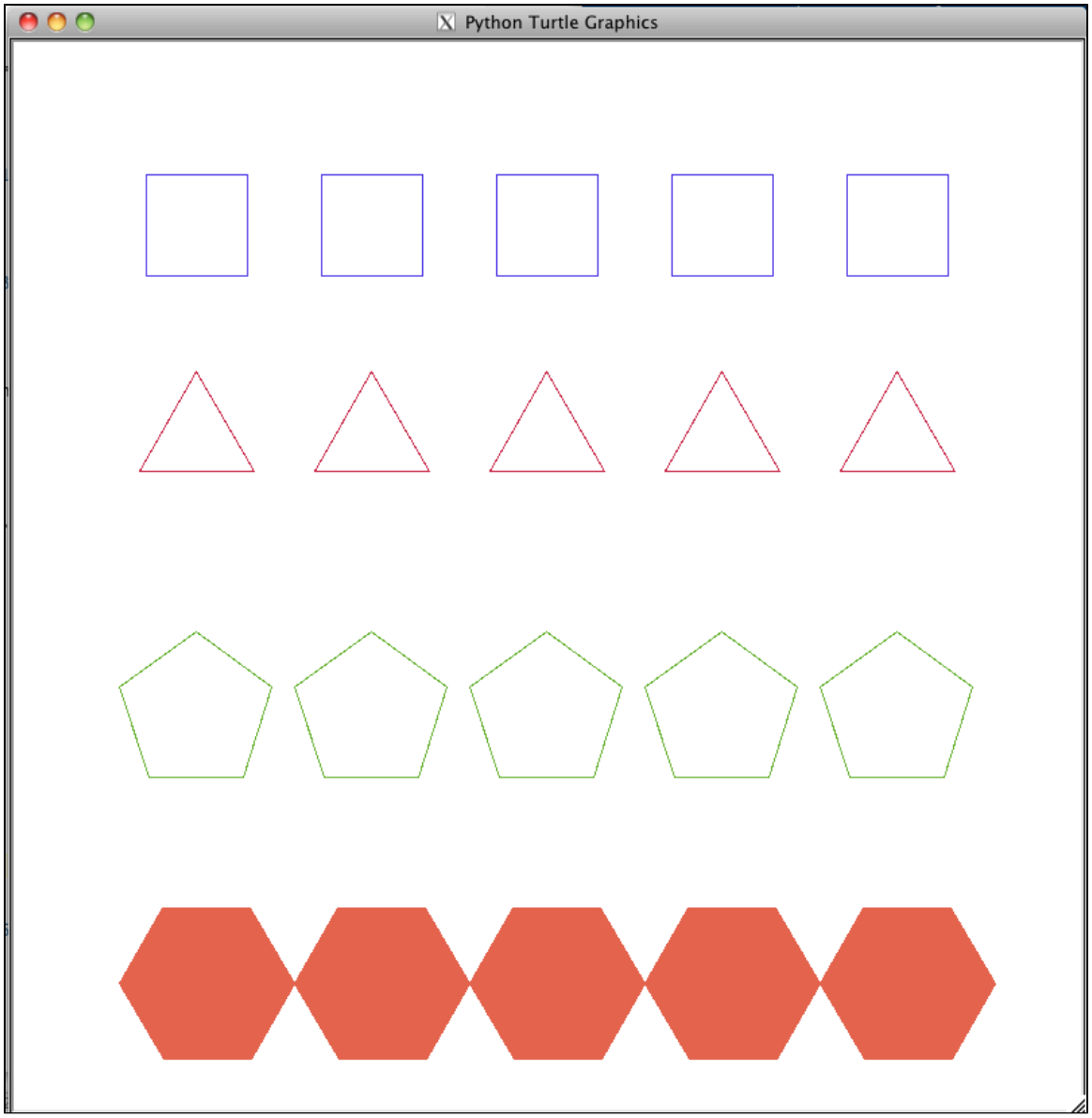
Lingar Project 9

In this project, I practiced using shape classes. I learned how to make classes inherit aspects of other classes. Specifically, I started out by writing a new class for a tree, which is an lstring of commands for drawing read in from a file. This tree class inherited information from a shape class, which contains the draw commands itself. In the init function for the tree class, I included a field which stated that the lsystem file would come from the command line. In lab, we had created a dictionary which allowed different rules to be used for the same lsystem. So, when I created a for loop to draw trees in a test method, each of them looked slightly different:



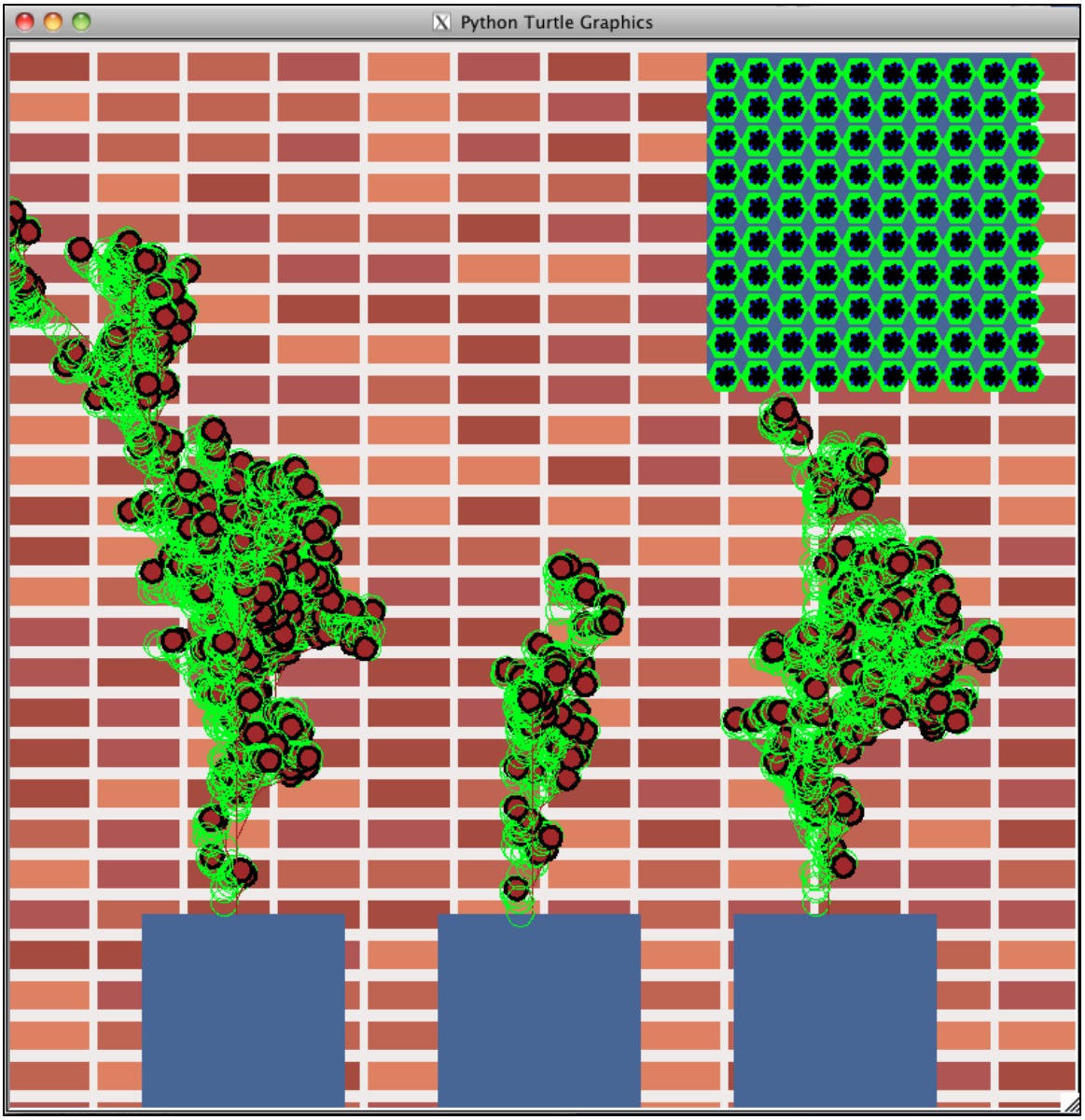
I made two new shape classes like the square and triangle. Mine are a pentagon and hexagon. I created a test function in the shapes file which creates a grid of the four shapes. Shapes change by row. I used for loops for each of these single shape rows. In the hexagon, I utilized the curly brackets which turn fill on and off.



I created an indoor scene. The wall is brick, and is formed from a new rectangle object I put in the shapes file. I called the object in a for loop, and created a list of random brick colors to be called in the for loop as well. I spaced the bricks based on a constant offset and their place in the for loop in order to allow the cement to show:

```
for i in range(20):
    for j in range(30):
        r.setColor( random.choice( color ))
        r.draw(-400+i*offset1, 390-j*offset2, scale=0.2, orientation=0)
```

I drew trees using my tree object. I put a tile from my mosaic (explanation below) on the wall as a painting.

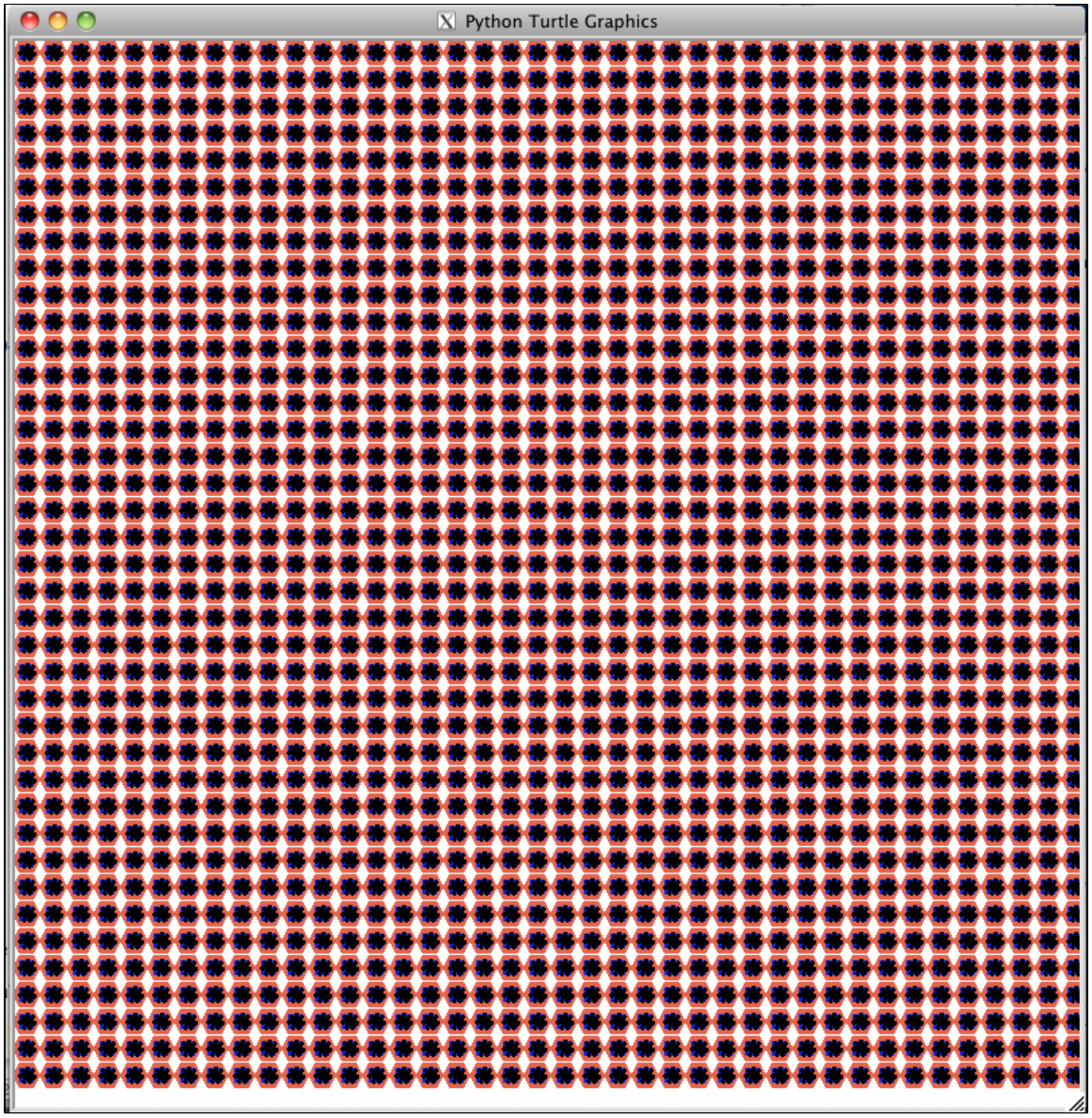


In order to create a mosaic, first I used a for loop to create a single 10x10 tile which could be scaled. This is in a function specifically for creating a tile:

```
for i in range(x):
    for j in range(y):
        h.draw(-400+i*offset, -370+j*offset, scale*0.005, orientation=60)
        s.draw(-400+i*offset+0.26*scale, -370+j*offset+0.26*scale, scale*0.005, orientation=0)
        for k in range(8):
            r.draw(-400+i*offset+0.54*scale, -370+j*offset, scale*0.001, orientation=50*k)
```

This loops over columns and rows, placing one hexagon, one square, and 8 rectangles in each spot. The position of each object is dependent on offset, position relative to the rest, and scale in order to make the tile scalable as a whole.

Next, I made a mosaic function, in which a quadruple for loop places tiles as above in a 5x4 grid. I added variables in order to make the tiles be properly offset, using variables for rows and columns of tiles.

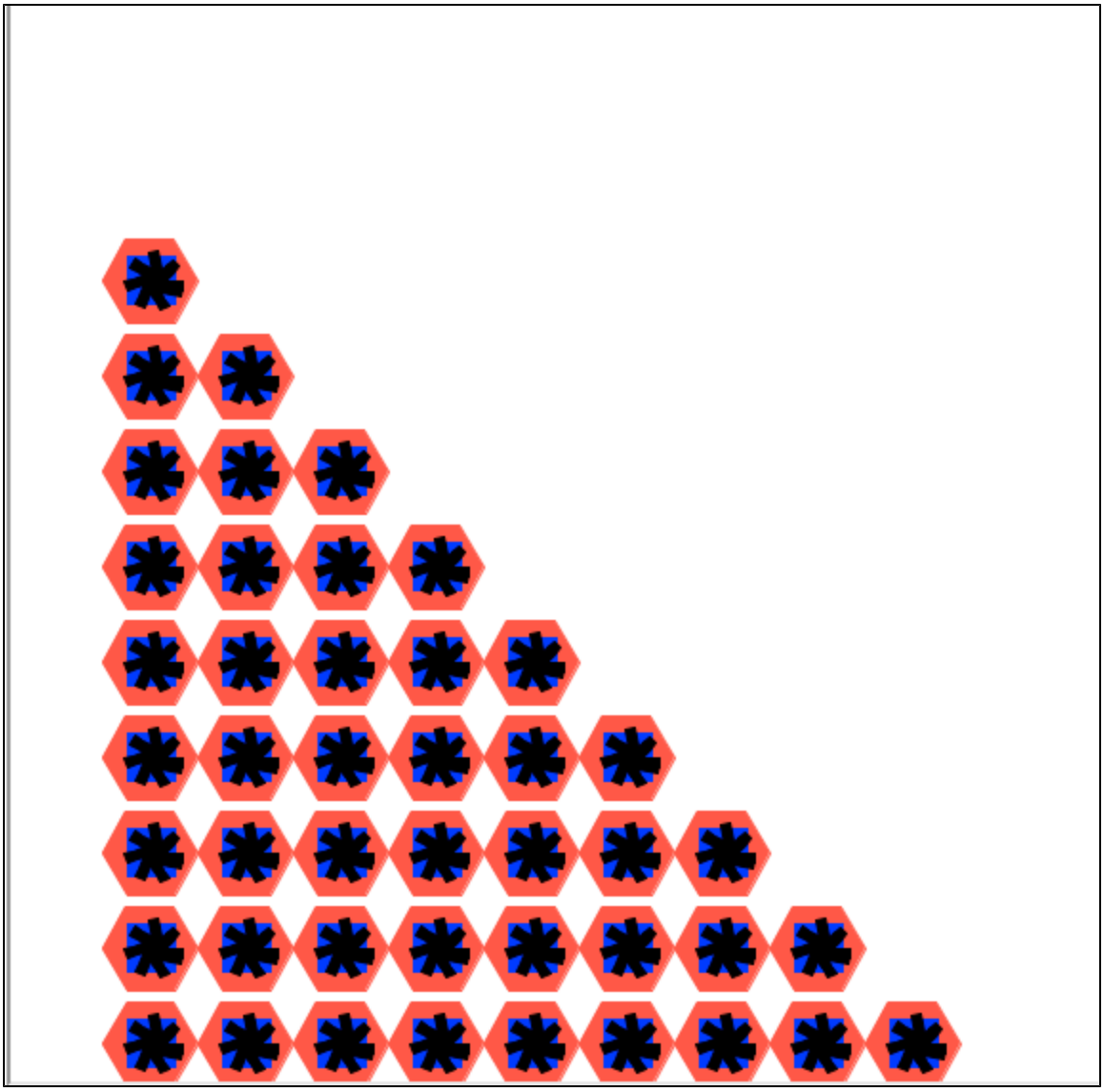


I learned how to make shapes classes inherit aspects of other classes, as well as how to overwrite specific portions of classes. I created images which are based on both lsystems and non-changing stings which are typed into code.

Extension 1: I made a triangular tile. I did this by creating "if" statements for each possible value of x, and subtracting the value of x from the value of y for each column. i.e.:

```
if i == 4:
    for j in range(y-4):
```

Here is the triangle:



Extension 3: I made droopy branches by changing the angle in the tree object to be 30, rather than 22.5