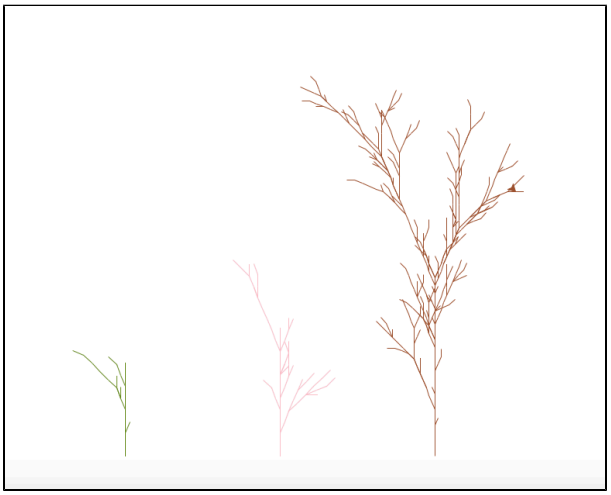
Project9

In this project, we continued to develop our knowledge and skills with making classes and working with inheritance. We used a Shape class with multiple sub-classes within in it that created different shapes. Also, we in this lab we created a tree class. To do all of this, we had to carefully work with inheritance from class to class and from file to file. Overall, the goal of this lab was to create a collection of shapes and images using classes and then placing these images/shapes into a scene as well as creating a mosaic of the images/shapes using loops, scaling, and positioning.

Task 1 asked us to make a new file called tree.py. In this file, we created a Tree class derived from the Shape class from Lab9, but with minor changes. Also, in this Tree class we created a few other methods called setIterations() and a new draw() method for this file. Here is a snippet of my code from tree.py:

```
class Tree(s.Shape):

    def __init__(self, distance=5, angle=22.5, color=(0.5, 0.4, 0.3),
            iterations=3, filename=None):
        s.Shape.__init__(self, distance, angle, color, istring = '' )
        self.iterations = iterations
        self.lsys = ls.Lsystem(filename)

    def setIterations(self, iterations):
        self.iterations = iterations

    def read(self, filename):
        self.lsys.read(filename)

    def draw(self, xpos, ypos, scale = 1.0, orientation = 0):
        terp = it.TurtleInterpreter()
        terp.place( xpos, ypos, orientation )
        terp.color(self.color)
        self.string = self.lsys.buildString(self.iterations)
        terp.drawString(self.string, self.distance*scale, self.angle)
```

Then we had to create a test() function to run the Tree class. First, we had to create the Tree object, then we could set its iterations, color, and finally draw the tree. Here is what an example of the code would have to look like:
"
t1 = Tree(filename = filename) t1.setIterations(3) t1.setColor('OliveDrab')
t1 = Tree(filename = filename)
t1.setIterations(3)
t1.setColor('OliveDrab')
t1.draw(-200, -300, 3, 90)
"
Then, we had to run the test code and produce three separate trees with different characteristics. To do this, I used a Lsystem from the CS Site that had multiple replacements for one rule.(systemJ.txt) The image produces was as follows:
Required Image 1:



Task 2 then asked us to create three new sub-classes of the Shapes class in our shapes.py from Lab9. We were asked to create at least three more shapes and their classes. I create a class for a star, diamond, cross, and pentagon. Here is a snippet of my new classes code:

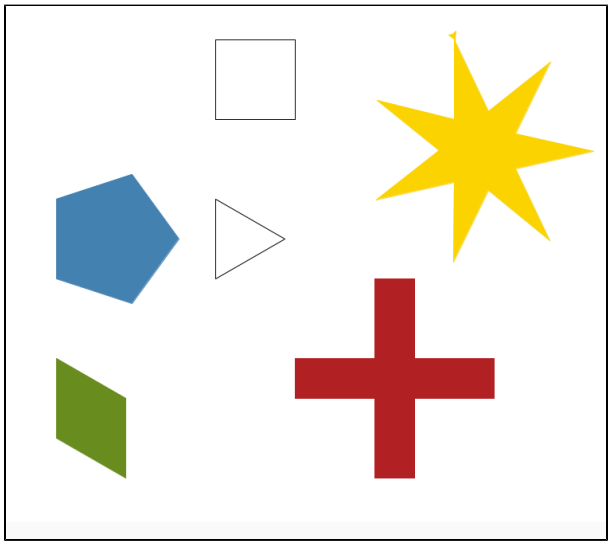
```
class Diamond(Shape):
    def __init__(self, distance = 100, color = (0,0,0) ):
        Shape.__init__(self, distance, 60, color, '{F--F-F--F-F--}')

class Star(Shape):
    def __init__(self, distance = 100, color = (0,0,0) ):
        Shape.__init__(self, distance, 51.5, color,
                       '{F---F++F---F++F---F++F---F++F---F++F---F++F---}')

class Cross(Shape):
    def __init__(self, distance = 50, color = (0,0,0) ):
        Shape.__init__(self, distance, 90, color, '{FF-F-FF+FF-F-FF+FF-F-FF+FF-F-FF}')
```

Then, another test code very similar to the tree.py test function was needed to draw the shapes. Here is a snippet of the test function code:

```
def test():
    t = Triangle()
    s = Square()
    p = Pentagon()
    p.setColor('SteelBlue')
    d = Diamond()
    d.setColor('OliveDrab')
    st = Star()
    st.setColor('Gold')
    c = Cross()
    c.setColor('FireBrick')
    t.draw(0, 0, 1, 90)
    s.draw(0, 200, 1, 90)
    p.draw(-200, 0, 1, 90)
    c.draw(200, -100, 1, 90)
    d.draw(-200, -200, 1, 90)
    st.draw(300, 200, 1, 90)
```

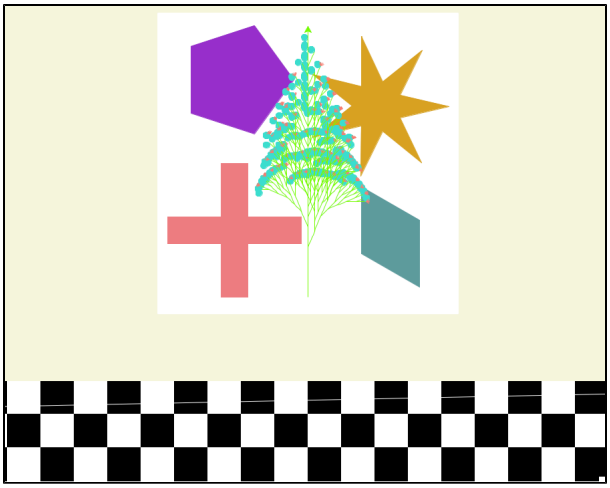
Here is the resulting image from running the test code for the shapes.py file:

Required Image 2:



Task 3 asked us to create a new file called indoorscene.py that created a scene that contained a painting of a collection of our shapes and trees. I decided to create a museum type scene with one large picture hanging on the wall. The main code for this was very similar to the previous two test code just with more specific scales and positioning. Here is my indoorscene.py image:

Required Image 3:

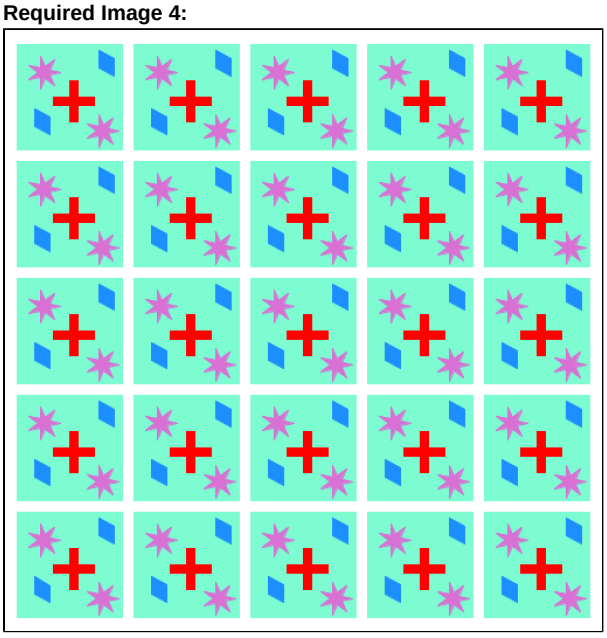


Task 4 then asked us to create a python file titled mosaic.py that created a mosaic of some of our shapes from our shapes.py file. To do so, we had to adapt the scaling of the background tiles and then create function that looped through the rows and columns of the image. Here is a snippet of code from the double loop code:

```
def mosaic(x, y, scale, Nx, Ny):
    for i in range(Nx):
        for j in range(Ny):
            tile(x+100*i, y+100*j, scale)
```

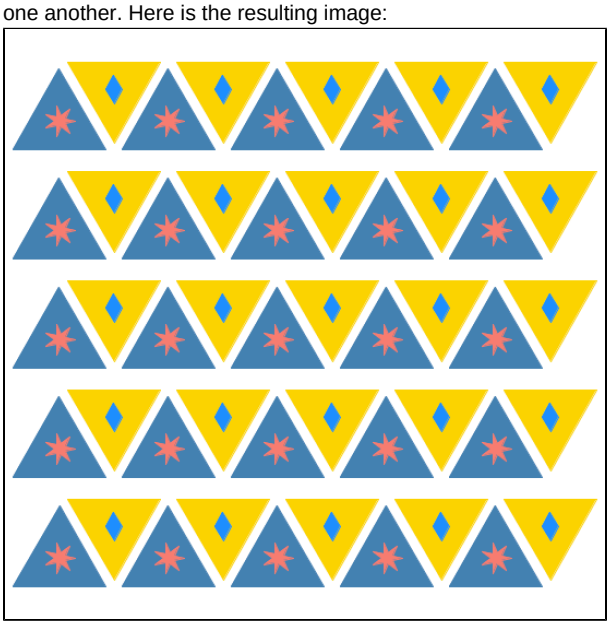
The resulting image was a mosaic with the scales, colors, positions, and orientations of my choice/input:

Required Image 4:

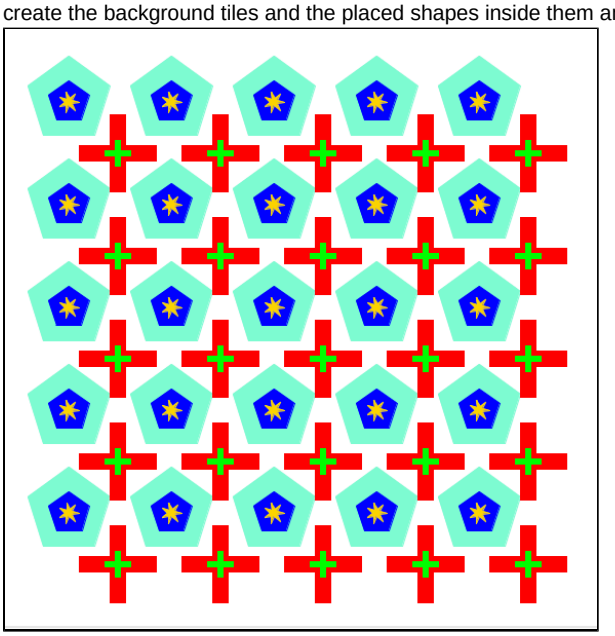


After I completed the four required task for this project, I decided to complete two extensions. I had a lot of fun creating and personalizing the mosaic file. So, for my extensions I completed one of the suggested extensions in two separate ways. For **extension 1,** I created a mosaic with

triangular tiles instead of square tiles. To do so, I had to create two triangle rows and shift the orientation and positioning of one so they fit inside one another. Here is the resulting image:



Extension 2, was just another version of the mosaic using two different shapes to create the tile background. I used pentagons and crosses to create the background tiles and the placed shapes inside them and customized the parameters. Here is the extension 2 resulting image:



What I learned. I furthered my knowledge of the class system. I gained a better understanding how to use classes to do create images, scenes, and mosaics. I also worked on my parameter skills will positioning, scaling, and orienting. In addition, we used more looping, turtle commands, and coloring. Overall, I feel like I better understand classes and how they work and how the computer stores and moved their information. In the end, I feel much better about classes and how to use them for creating images and scenes.

People who helped me. I received help from Professor Maxwell and Mike the TA. I worked alongside fellow coder Julia Saul.