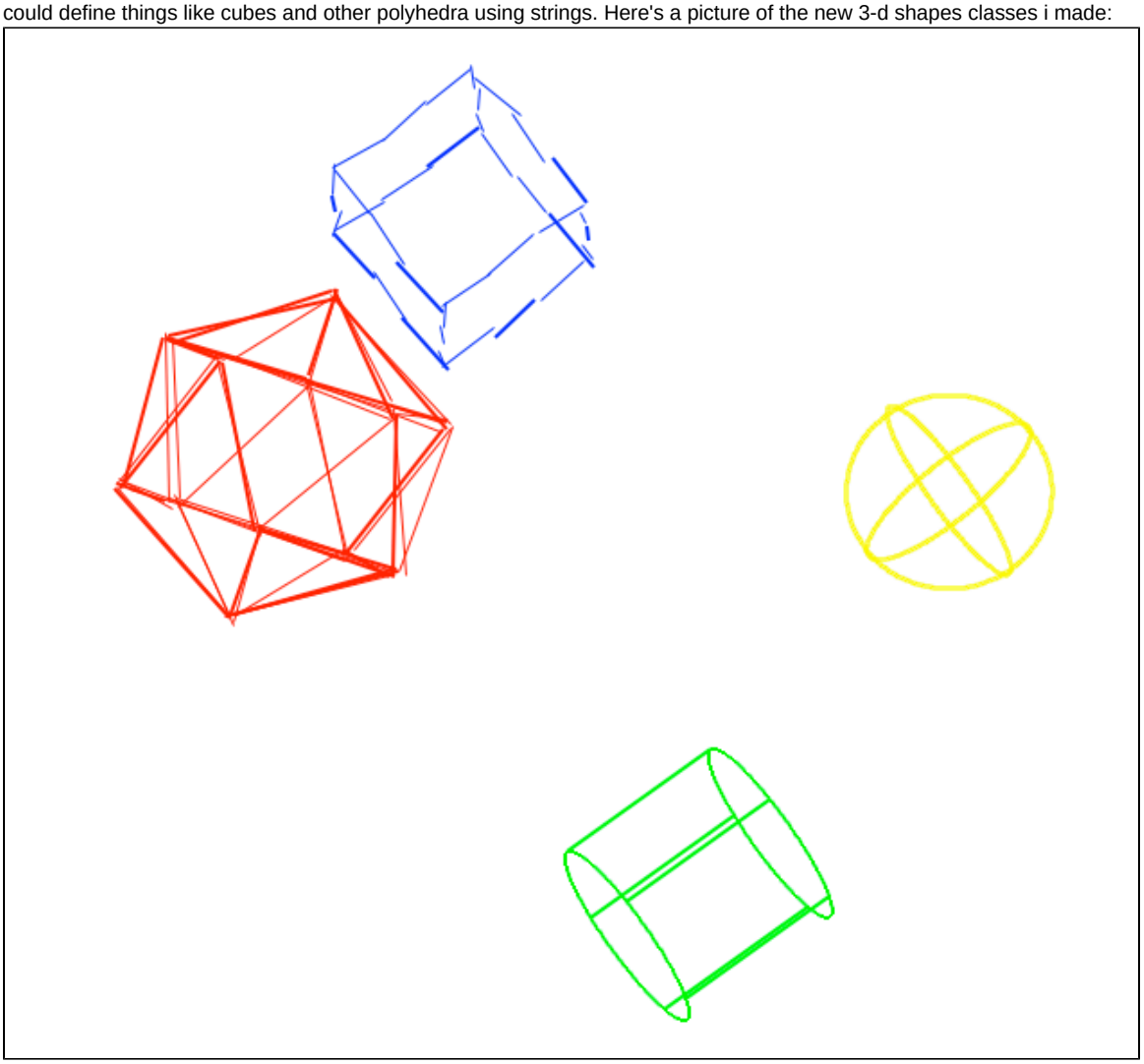
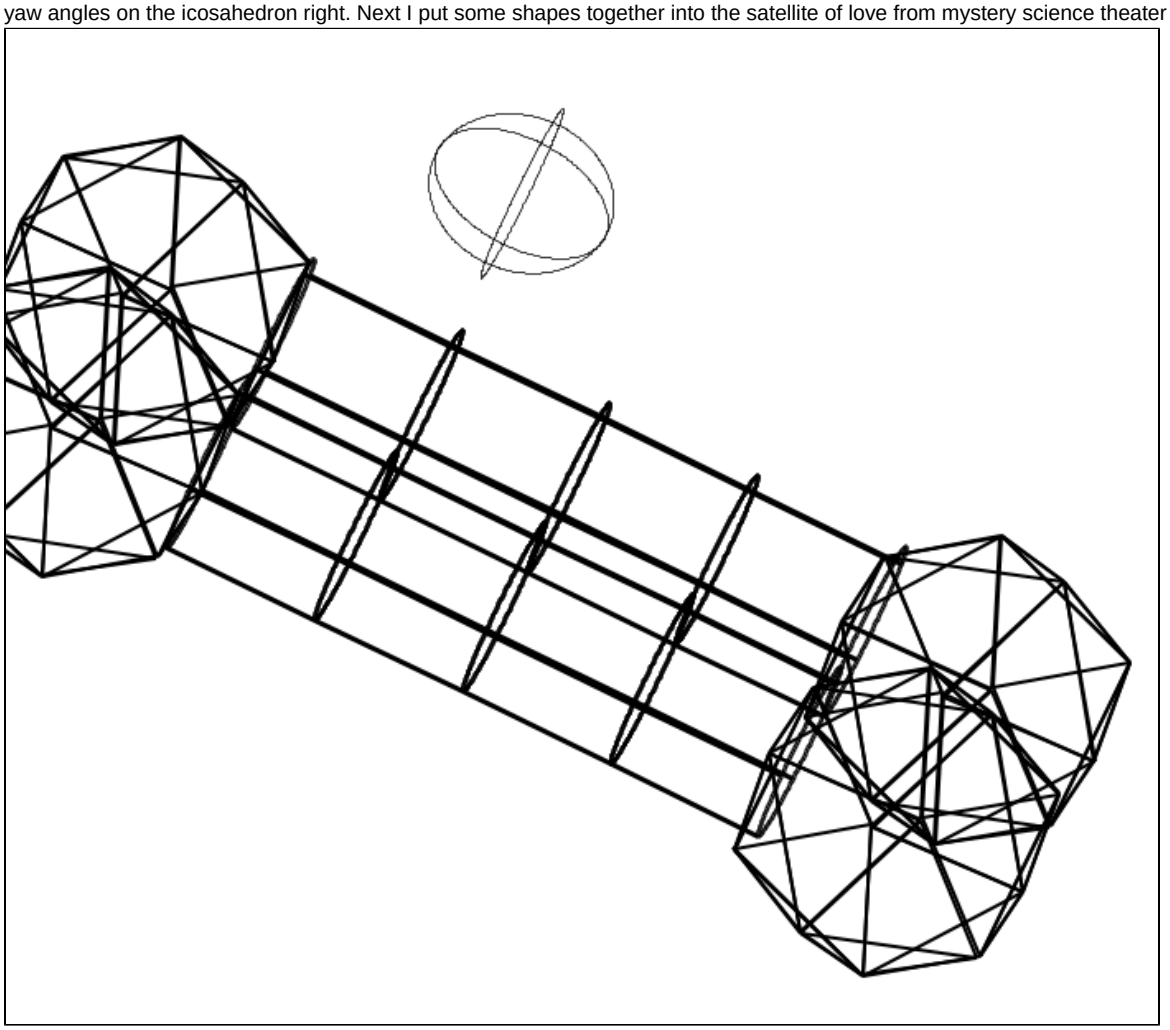
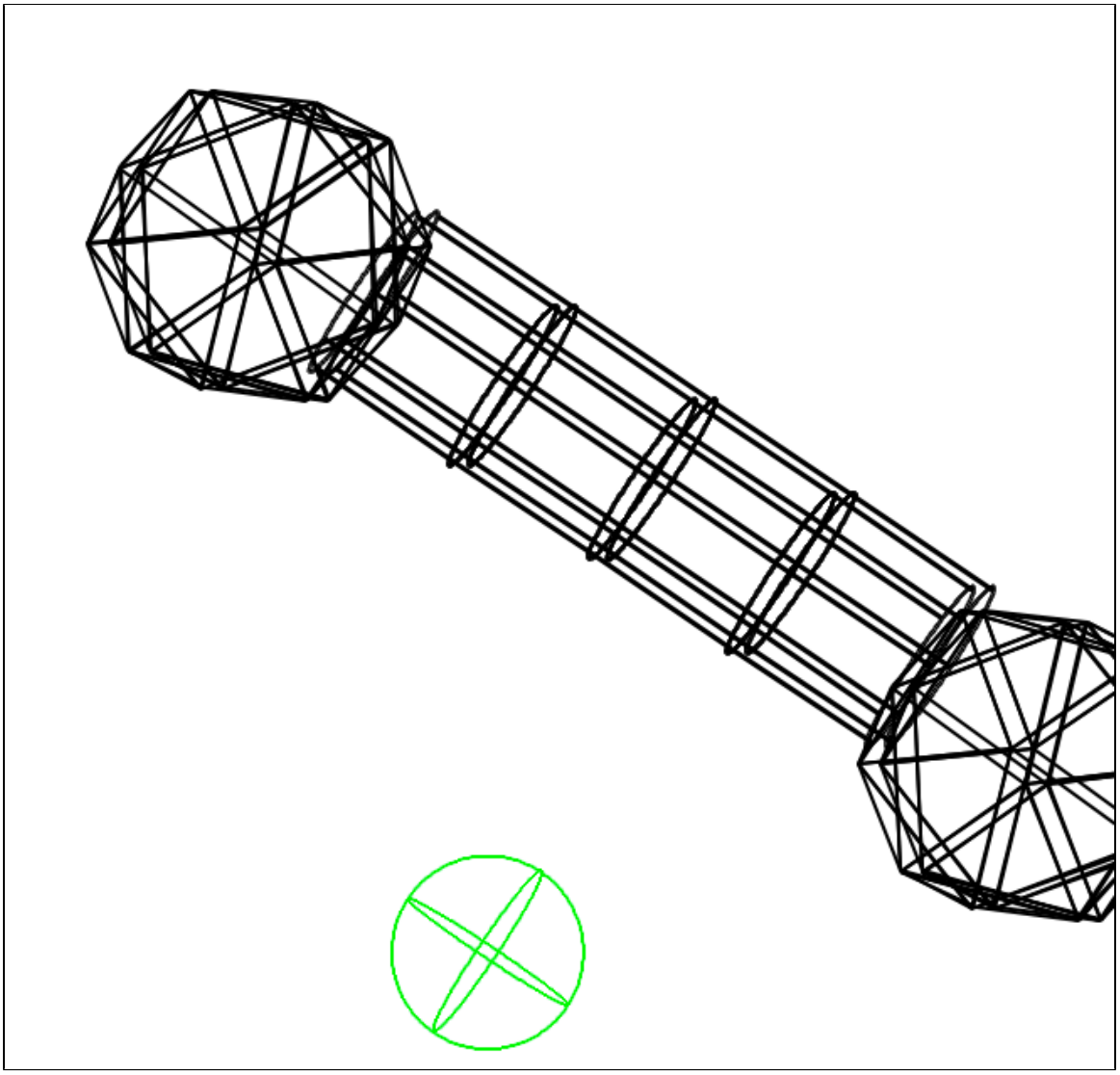
3d turtle

In this project we built on our graphics code from the last few weeks. This time we used the 3d turtle package to make our drawings in 3-d, using another, z, coordinate and two more orientation scalars, pitch and yaw. We had to make new methods in the turtle interpreter file to instruct turtle to move in the third dimension. Then we added symbols to the l-string reading if structure, to account for turns in the third dimension. Then we could define things like cubes and other polyhedra using strings. Here's a picture of the new 3-d shapes classes i made:



I made the icosahedron, the 20 sided regular polygon, along with a cube, sphere, and cylinder. The hardest part was getting all the right pitch and yaw angles on the icosahedron right. Next I put some shapes together into the satellite of love from mystery science theater 3000:





For my final extension I attempted to set the right click to make all of the shapes change to a random color. To achieve this I first had to make a function, colorSwap, that loops through the shape objects, changing their color and redrawing them in the same position.The tricky part was looping over a list of objects. I looked this up on stack overflow and copied some code from there, namely the a metaclass structure. Here's the code:

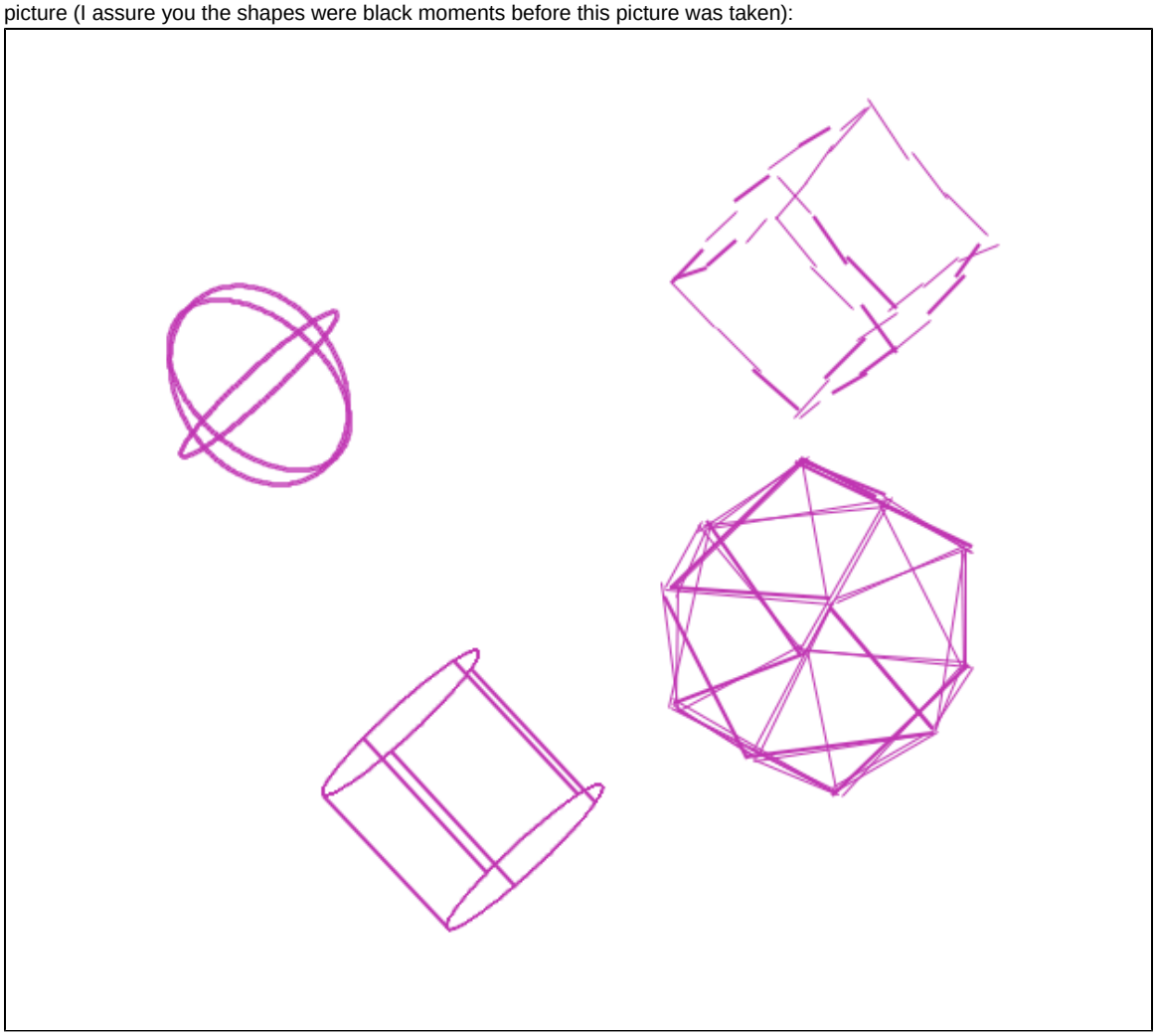
```
 class IterRegistry(type):
    def __iter__(cls):
        return iter(cls._registry)

class Shape(object):
    __metaclass__ = IterRegistry
    _registry = []

    def __init__(self, name):
        self._registry.append(self)
```

Then I could loop through each shape object and redraw it the same places I drew it before.

To record the shape's position I added a stack to the place where the shapes are first drawn, and added the drawing position to the stack. Then when it came time to redraw the shape I popped its location(s) off the stack to redraw them in the same place. Andrew Fletcher helped me make sure this stack emptied out properly each time the shapes are redrawn. He also told me how I need to pass the object list to Colorswap in the

function call, with a special name, because it would not work otherwise.

I tried to make colorSwap work on a right click but this proved to be a tenacious problem because the symbol table where the shapes objects were stored was not accessible to the turtle interpreter, as Fletcher helped me to see. Unfortunately I couldn't solve the problem of making the objects change color on a right mouse click, but my colorSwap function did work when called directly in a test function, as you can see in this picture (I assure you the shapes were black moments before this picture was taken):



In this project I learned that simple solutions to complex tasks are what computers are good for, so when things get very complicated, like drawing an icosahedron, using python might not be worth it.