Assignment 11 (Philip Prosapio)

Philip Prosapio
Spring 2010 - CS151

Note- My code for this project is in the pwprosap account on the academic server.

Summary

For this lab we were taught how to use the 3D turtle and were then asked to create multiple simple 3D objects for the first task, like a cube or pyramid, etc. Then we had to put these shapes together to create a simple 3D scene. The final task asked us to take a extension and complete it but then give a detailed description of how we completed it in this write-up.

Solution to Tasks

1) For the first task I created five different simple shapes. A cube, pyramid with a square base and a pyramid with a triangular base, a prism and a rectangular box. To help us make sure that these objects were created correctly and that they were oriented right we were able to use some code to create a coordinate axis to help us is the creation process. Here is the code for the axis:

def drawAxes():

1. Draw the axes (y in blue, x is green, z is red)
2. y-axis
   ln = Line(distance=300, color=(0,0,1))
   ln.setWidth(5)
   ln.draw(0,0,zpos=0, roll=0, pitch=0, orientation=90)
3. x-axis
   ln.setColor((0,1,0))
   ln.draw(0,0,zpos=0, roll=0, pitch=0, orientation=0)
   #z-axis
   ln.setColor((1,0,0))
   ln.draw(0,0,zpos=0, roll=0, pitch=90, orientation=0)

The picture of all of my shapes, but not the axis, is attached and labelled picture1lab11.png .

2) For the this task I created a city street with two rows of houses on opposite sides of a street with a sun in the sky. For this task I had to make sure all of my objects had the correct dimensions so that they would fit together, an example is that I had to make sure my cube face was the same size as the bottom face of the square pyramid. I also decided to make a sun out of my pyramid shape. I used the for loop below to rotate the pyramid around an axis to make a sun like object.

```
for i in range(360):
p = Pyramid(1, 'yellow')
p.setStyle('jitter')
p.setWidth(1)
p.draw(-100, 250, zpos=100, roll=-90, pitch=01.0*i, orientation=0, scale = 0.5)
```

I took two pictures of this scene from two different angles and perspectives. They are attached and named picture2lab11.png and picture3lab11.png .

3) For the task of taking an extension and completing it I chose the extension that had me create a drawing style that I had not created yet and make it. I chose to make a drawing style that was was slightly based off of the dash style but just more advanced. I called it my 'crosshatch' style, in which instead of having individual dashes in the place of a line, I made it so each line was made up of perpendicular slashes that form X's. My approach was to keep the same code from the dash style to determine number of dashes but then I needed to make the code that would change the orientation and draw the two perpendicular slashes.

```
for i in range (numdash):
ori = turtle.heading()
pos = turtle.position()
turtle.setheading(60)
turtle.forward(self.dashLength)
turtle.left(180)
turtle.forward(2*self.dashLength)
turtle.setheading(ori)
turtle.goto(pos)
turtle.setheading(-60)
turtle.forward(self.dashLength)
turtle.left(180)
turtle.forward(2*self.dashLength)
turtle.setheading(ori)
turtle.goto(pos)
turtle.up()
turtle.forward(10)
turtle.down()
```

The key parts of this code that made the crosshatch style work were the two variables ori and pos. These two variables saved the position and orientation of the turtle each time it moves forward for each dash. The picture of a couple of my 3D shapes that are using the crosshatch drawing style is attached and labelled picture4lab11.png .

Extensions

Extension 1 - For this project I decided to do only one extension. I chose to create a binary tree made out of pyramids for my extension. By using recursion I was able to make a binary tree out of pyramids in which each pyramid is a different randomly chosen color. I placed children branches at each of the four bottom corners of the parent pyramid. Having my base case restrict how many times this code would run, I used recursion to make a ever expanding tree. Here is the key part of my code which is the original draw call, and the recursive call:

self.draw(xpos, ypos, zpos = zpos, scale = scale, roll = -90)

1. draw right back branch
   self.drawBinaryTree(xpos = xpos+self.distance*100*scale*1/4, ypos = ypos-self.distance*100*scale/2, zpos = zpos-self.distance*100*scale*1/4, scale= 0.5*scale)

The reason why the second part of this code is recursive is because I am calling the function within itself, so this technically could go on forever if not for my base case that I used:

1. if the scale is too little , then it is time to stop
   if scale < 0.1:
   return

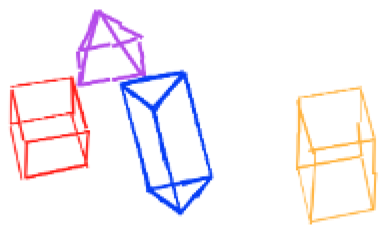
The the two pictures of my recursive binary tree from different angles are attached and labelled picture5lab11.png and picture6lab11.png .

What I learned

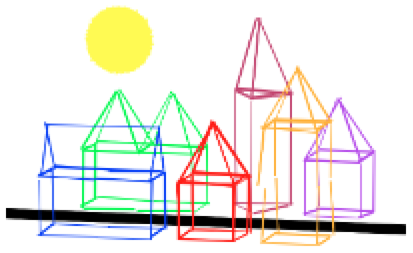
The most interesting thing I learned from this lab was how to use the 3D turtle and the concepts of the roll, pitch, and yaw. It is weird to think of drawing in three dimensions on a two dimensional screen, but it was very interesting to learn.

Attached Pictures

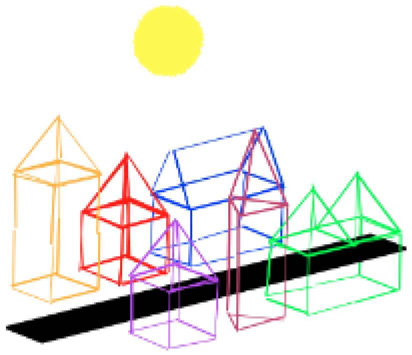
picture1lab11.png



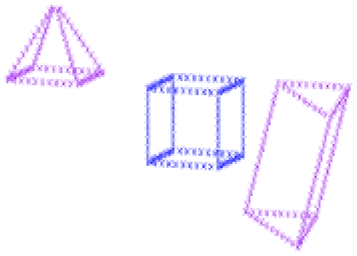
picture2lab11.png



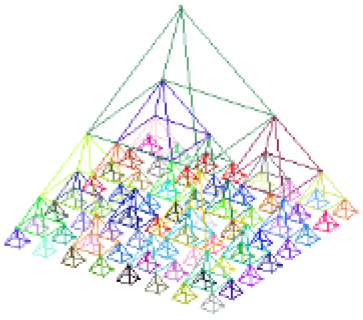
picture3lab11.png



picture4lab11.png



picture5lab11.png



picture6lab11.png