CMSC 330 Project 1

Burkhardt, Ryan

UMGC CMSC 330

Author Note

This program reads in a file and creates a UI of hollow and solid polygons after parsing words from the .txt file. This uses many different methods of Object-Oriented Programming as well as Lexer errors and Syntax errors.

UML Class Diagram

A diagram of a program

Description automatically generated

**Project 1**

For project 1 I started by looking over all sent files and tore them apart to understand the complexity of the assignment. The lexer class was completely new and it took a good bit to understand how it was working and then reading through the text it helped me realize it was parsing the selected input as strings from the built-in enumeration Token class. This was where I started by inputting all the tokens required for the Isosceles Triangle, Regular Polygon, and Parallelogram, and Text. After all the required Tokens were in, I started working with the Text class. This started out by reading and following the provided instructions. I went by creating the string and point for the text to appear on the GUI in reference from polygon\_. I used this to extend the image class like polygon\_ and then used a similar draw method but with a point and a string. Also, the graphics constructor is called and used to form the colorDrawing and then attaching the string using graphics.drawString method.

Second, was the constructor for the parallelogram. I found this one to be the simplest because when looking at it I could logically think of what the offset was doing. It visually made sense to me and I could use the methods to alter rectangles points using the offset. This is when I hit a large snag with a second point. I could not figure out how to get the second point from the AT method. I kept getting an expected comma, then after adding COMMA to token it expected right parenthesis. After removing these from the beginning where I initialized the rest of my variables such as offset and numsides. I then put it in the actual if statement for the parallelogram and the error went away. I then put in a right parent parameter and could also remove the error. This was something that confused me and I was left not fully understanding the error. With this implemented the Text was properly working. But I can also use right paren token to make it work.

The third class of the project was the isosceles triangle. This was identical to the right triangle, but the x and y values had to be manipulated slightly. Instead of starting in the upper left we start at the top center. Then we just modify the x values to be half the width of either direction after subtracting the height of the center point. Then these two points will have the same y value and the split x value. Other than those changes it also extended the solid polygon and then it was identical in the vertices and the call createPolygon method.

Lastly, I worked on the polygon class to conclude the project. This was a great challenge because the calculation was something I was not familiar with. Taking on the task I went into my geometry searches and found a reference from (Deinst, 2010). While in C++ I believe the formula he used the radius to multiply radius + cosign (2pi \* i/n). This however needed a slight modification to include the x and y coordinate to not be centered in (0,0) this is when we would have to add this after the calculation is complete so we could have the appropriate shift take place afterwards. Using this formula, I was able to calculate the coordinates of each triangle which would give me my point to construct the polygon. This in turn led to the creation of empty lists and a for loop to add each point to a list. I then ran into a constructor error in drawpolygon because it could not take a double as an input. So, I then converted it into an integer before adding to the x\_points list. This concluded the polygon as an extension to the solid polygon class and produced accurate results.

In conclusion I have found this to be one of the most challenging yet most rewarding GUI applications I have worked on. It has also been in my opinion one of the only projects that I can relate to how the outside career would entail. I believe working on another person’s source code to figure out not only how there code is working but, to also figure out how to implement your own code inside was very close to how working in a professional environment would be. In this project I have learned a lot about how the lexer parsing works based off the enumerate function of tokens as well as how to merge my own code into an existing project. It was both a challenge and a reward to have this project working.

**References**

**Deinst. (2010, August 8). Calculate coordinates of a regular polygon’s vertices. Stack Overflow. https://stackoverflow.com/questions/3436453/calculate-coordinates-of-a-regular-polygons-vertices**

**CMSC 330 Project 1**

**Test Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case** | **Input** | **What to Test** | **Actual Output** | **Pass/Fail** |
| 1 |  |  |  | Pass |
| 2 |  |  |  | Pass |
| 3 |  |  |  | Pass |
| 4 |  |  |  | Pass |
| 5 |  |  |  | Pass |

**Screenshots**

**Test Case 1**

**Test Case 2**

**Test Case 3**

**Test Case 4**

**Test Case 5**