Antonio Scalfaro

CMSC 330

Professor Yuhua Jiang

Project 1

Due: 04/09/2024

**Approach:**

The approach to this project took some time to formulate. This project was decisively different from projects I have completed in the past and I had to take a few hours to gather my footing. I had never made any parsing/lexer projects before, and I had to understand what was being asked of me before implementing anything. I tackled the Text class first, thinking that it may be the easier of the classes to implement, as it turns out it was the hardest of the classes to implement. This was due to the changes that had to be made to the lexer and the parser files. I had to refactor the lexer file to handle strings, no other class presented this particular challenge. Once I finished the lexer, I moved to the parser and followed the same structure of the existing code. After the text class was made, the lexer and parser refactored, I added the additional test for text into the scene test file. I tested to make sure it worked and after a few tweaks it worked just fine. The rest of the classes went very smoothly after that. I followed the same approach: create the class, refactor the parser (no more changes to the lexer were necessary), add the new test to the scene test file, and test to ensure it worked properly. I was intimidated by this project at first but was pleasantly surprised that I caught on quickly and had very little issues implementing the desired functionality.

**Test Plan:**

The testing of this project is all done through one file, the scene.txt test file. This file includes the language to implement all of the shapes and text graphics in the program. Since this is a GUI program, there are no physical inputs from the user to test, I just ensured that the program utilized the test file and all graphics were correctly input into the GUI. I ensured to test after every new implementation so that I could work out any kinks that may have arisen, doing this one at a time was much easier than testing them all at once when I would have to dig through the errors to find what particular piece was going wrong. I do note, however, that my final output looks slightly different than the provided output in the project 1 file. The parallelogram that is displayed in the given output looks different than the parallelogram that is displayed in my program. I double and triple checked how I calculated the points of the parallelogram and am confident that I have done so correctly. The points given for upper left and lower right are (340, 50) and (440, 120) respectively with an x-offset of 30. This would make the four points starting from upper left moving clockwise to the lower left as follows: (340, 50), (370, 50), (440, 120), (410,120). These values do not appear to be the same as the parallelogram displayed in the given output. Even with this small discrepancy, I believe that my program provides correct output and has passed all the testing requirements. There is a second file, sceneSyntaxErrors.txt, that can be used to check that incorrect syntax is caught still. I did not really know what was expected of the rubric requirement other than to include a scene file that could not be displayed due to syntax errors.

**Test Output:**

A screenshot of a computer

Description automatically generated

**Lessons Learned:**

While most of the coding for this project was nothing new, the challenge of understanding and refactoring a lexer and parser certainly was. I learned how to examine these types of files to understand how they were working and to implement changes to add additional functionality to them. As I said previously, this was a daunting project at first but after a few hours of getting my footing, I was on a roll and finished coding it a few hours after that. I am not extremely familiar with java’s awt library so there were some growing pains there too, but none that couldn’t be overcome by examining the existing code of the other classes and following their layout. Ultimately, this was a fun project to work on and expanded my knowledge base. If I were to add more to the functionality of the program, I may try to implement some kind of mouse events in the GUI. My first thought was to be able to remove shapes by right clicking on them and to add additional shapes by adding points to the GUI using left clicks and using a submit button that would produce a shape based on the number of points added to the GUI. These may be fun challenges to try on my own in the future.