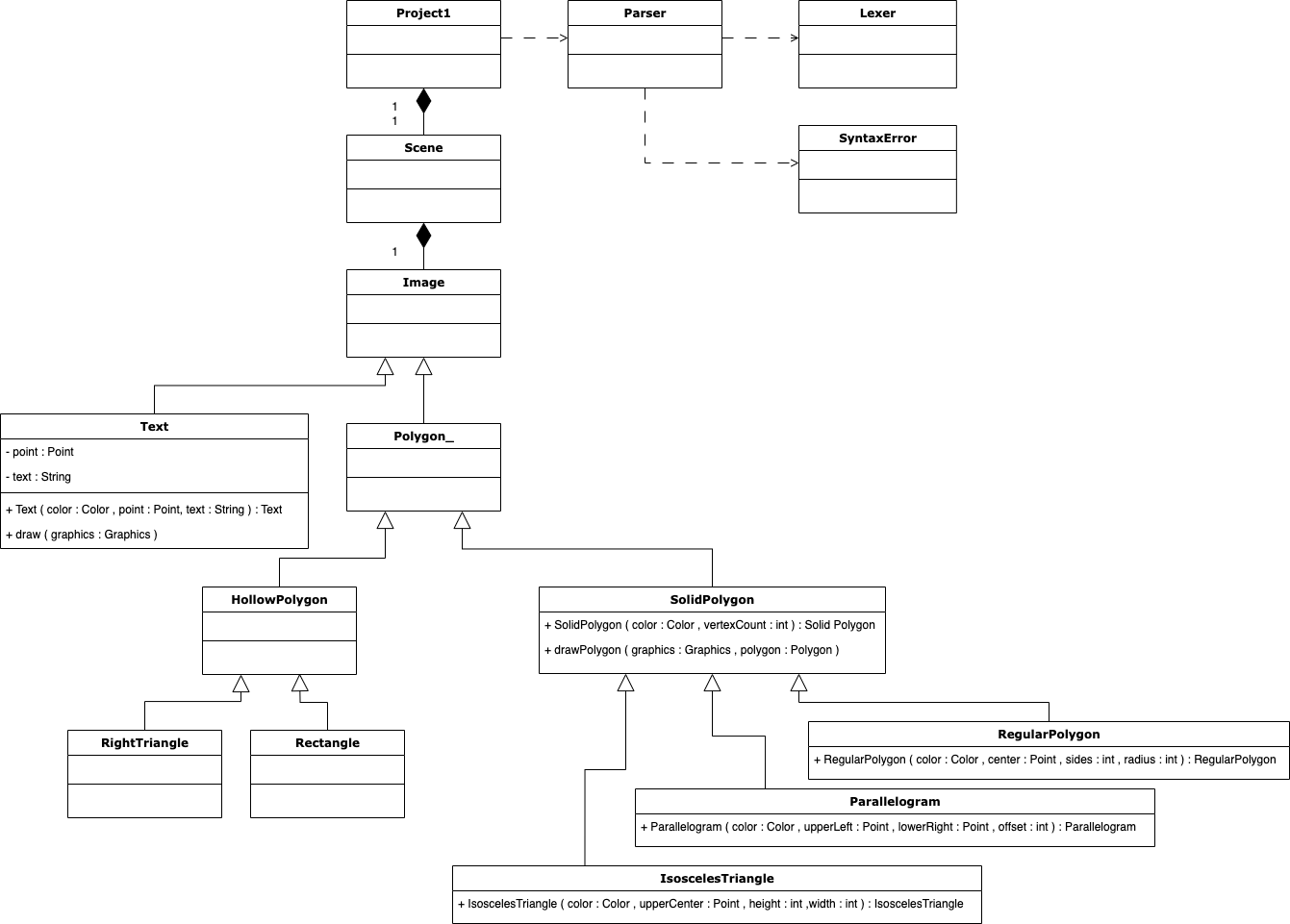
Project one extenstions were four images: parallelogram, regular\_polygon, isosceles, and text. This involved creating classes for each image, and an additional class, SolidPolygon.

My goal was to first create the output that was provided in the Project 1 instructions. I approached the project by reviewing the code and figuring out how the classes were interacting. I knew that I needed to add additional required tokens and create five more classes, so I completed those first. I then moved on to the Parser class and added the required productions. For the additional classes, I referenced existing classes and modified them (ie. HollowPolygon referenced to create SolidPolygon, changing drawPolygon to fillPolygon). The isosceles triangle was the easiest, the regular polygon was the hardest mathematically. I ended up finding an equation to reference for the regular polygon on stack overflow. The instructions asked me to modify the Lexer class to handle string tokens, but I found that I didn’t need to. Below is a UML class diagram for the new classes as well as some test cases. I added content for the new classes in the UML diagram.

Regular Polygon referenced from: <https://stackoverflow.com/questions/25945593/calculate-6-vertices-of-randomly-generated-hexagon>



|  |  |
| --- | --- |
| Test 1: Skeleton Tiles - PASS | |
| Input Values | Scene Polygons (500, 500)  RightTriangle Color (255, 0, 0) at (50, 30) Height 100 Width 300;  Rectangle Color (0, 128, 255) at (100, 100) Height 200 Width 100;  Isosceles Color (255, 0, 0) at (120, 120) Height 100 Width 200;  Parallelogram Color (0, 0, 255) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (255, 0, 255) at (300, 300) Sides 6 Radius 80;  Text Color (0, 0, 0) at (400, 200) "Hello World";  End. |
| Expected Output | Skeleton tile images for hollow right triangle and rectangle defined in parser. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 2: Add IsoscelesTriangle - PASS | |
| Input Values | Scene Polygons (500, 500)  RightTriangle Color (255, 0, 0) at (50, 30) Height 100 Width 300;  Rectangle Color (0, 128, 255) at (100, 100) Height 200 Width 100;  Isosceles Color (255, 0, 0) at (120, 120) Height 100 Width 200;  Parallelogram Color (0, 0, 255) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (255, 0, 255) at (300, 300) Sides 6 Radius 80;  Text Color (0, 0, 0) at (400, 200) "Hello World";  End. |
| Expected Output | Added image utilizing SolidPolygon and IsoscelesTriangle. Expected output is a red isosceles triangle with input values mentioned above. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 3: Add Parallelogram - PASS | |
| Input Values | Scene Polygons (500, 500)  RightTriangle Color (255, 0, 0) at (50, 30) Height 100 Width 300;  Rectangle Color (0, 128, 255) at (100, 100) Height 200 Width 100;  Isosceles Color (255, 0, 0) at (120, 120) Height 100 Width 200;  Parallelogram Color (0, 0, 255) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (255, 0, 255) at (300, 300) Sides 6 Radius 80;  Text Color (0, 0, 0) at (400, 200) "Hello World";  End. |
| Expected Output | Added image of parallelogram. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 4: Add RegularPolygon - PASS | |
| Input Values | Scene Polygons (500, 500)  RightTriangle Color (255, 0, 0) at (50, 30) Height 100 Width 300;  Rectangle Color (0, 128, 255) at (100, 100) Height 200 Width 100;  Isosceles Color (255, 0, 0) at (120, 120) Height 100 Width 200;  Parallelogram Color (0, 0, 255) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (255, 0, 255) at (300, 300) Sides 6 Radius 80;  Text Color (0, 0, 0) at (400, 200) "Hello World";  End. |
| Expected Output | Added image of regular polygon. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 5: Add Text (completed given scene) - PASS | |
| Input Values | Scene Polygons (500, 500)  RightTriangle Color (255, 0, 0) at (50, 30) Height 100 Width 300;  Rectangle Color (0, 128, 255) at (100, 100) Height 200 Width 100;  Isosceles Color (255, 0, 0) at (120, 120) Height 100 Width 200;  Parallelogram Color (0, 0, 255) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (255, 0, 255) at (300, 300) Sides 6 Radius 80;  Text Color (0, 0, 0) at (400, 200) "Hello World";  End. |
| Expected Output | Add image of text, “Hello World” and complete the scene given in project 1 instructions. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 6: scene2.txt file - PASS | |
| Input Values | Scene Polygons (500, 750)  RightTriangle Color (0, 0, 255) at (50, 330) Height 100 Width 300;  Rectangle Color (255, 0, 0) at (100, 100) Height 200 Width 100;  Isosceles Color (0, 0, 255) at (520, 120) Height 100 Width 200;  Parallelogram Color (255, 0, 0) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (0, 0, 0) at (300, 300) Sides 6 Radius 80;  Text Color (250, 0, 250) at (265, 300) "Hello World!";  End. |
| Expected Output | I adjusted the scene.txt file and renamed it scene2.txt. The goal is to check the functionality of the program when you adjust colors, points, and sizes.   * Increase the scene width by 50%. * Switch colors of the three and four vertex shapes. * Move the isosceles triangle to the right, and move the right triangle down. * Print text “Hello World!” (added exclamation mark) inside the regular polygon. |
| Actual Output |  |

|  |  |
| --- | --- |
| Test 7: scene3.txt file - PASS | |
| Input Values | Scene Polygons (500, 750)  RightTriangle Color (0, 0, 255) at (50, 330) Height 100 Width 300;  Rectangle Color (255, 0, 0) at (100, 100) Height 200 Width 100;  Isosceles Color (0, 0, 255) at (520, 120) Height 100 Width 200;  Parallelogram Color (255, 0, 0) at (340, 50) (440, 120) Offset 30;  RegularPolygon Color (0, 0, 0) at (300, 300) Sides 6 Radius 80;  Text Color (250, 0, 250) at (265, 300) "Hello World!";  End. |
| Expected Output | I adjusted the scene.txt file and renamed it scene2.txt. The goal is to check the functionality of the program when you adjust colors, points, and sizes.   * Increased the sizes of the shapes * Made the rectangle into a square * Added two sides to the regular polygon. |
| Actual Output |  |

I started on Project 1 as soon as it was posted on March 8. A little overwhelming at first because JAVA is neither my first coding language or my favorite. I enjoy this class because it is helping me understand how these languages work, so dots are connecting on languages that I’m dabbling with and trying to understand better.

I learned a lot about how tokens are used and parse the different parts of the text code. The way the parser utilizes these tokens to pass to the individual classes was very interesting. Once I understood the code from Project 1, creating/extending the classes was straight forward. Drawing my own images was awesome. I hope to see more of that in CMSC 405 Computer Graphics later this year. Overall excited to tackle the project, pleased with the outcome, and looking forward to the C++ project (where I have zero experience).