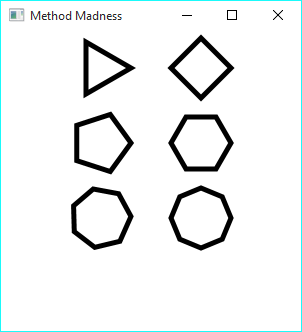
Computer Science is a new frontier. Historically, it is a somewhat recent discovery. It provides new challenges and obstacles which we must overcome. Computer Science opens up a whole world of possibilities, pushing the limits of what is possible. I have attempted to experience just a fraction of this grand opportunity with my meddling in JavaFX. I have created a program that uses mathematical principles to produce visible output in the form of shapes, patterns, and colors.

The wonderful world of mathematics allows us to perceive and affect the universe around us in extraordinary ways. Computer Science is deeply rooted in mathematics. Therefore, I have utilized mathematics in my Java program. My program calculates interior angles of various regular polygons in order to obtain the x and y coordinates of that angle based on trigonometric ratios. For example, when an instance of the *RegularPolygon* class is created, the coordinates of the vertices of the desired regular polygon are instantly calculated and set within that instance.

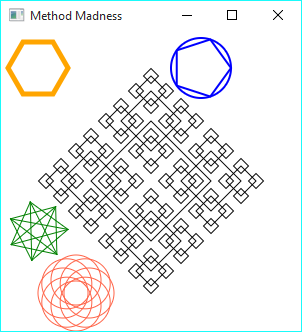
|  |
| --- |
| void setCoords() {  double x;  double y;  for (int i = 0; i < numberOfSides; i++) {  x = centerpoint[0] + (radius \* Math.cos(Math.toRadians(interiorAngle \* (i + 1))));  y = centerpoint[1] + (radius \* Math.sin(Math.toRadians(interiorAngle \* (i + 1))));  coords\_x[i] = x;  coords\_y[i] = y;  }  } |



This way, I am able to create any kind of regular polygon. Polygons are created using the *drawRegularPolygon()* method. The method takes as inputs a *GraphicsContext* object, a *RegularPolygon* object, a *Color* object, and an integer value specifying line width. The values stored in an instance of *RegularPolygon* determine the type of polygon, location and size of the polygon drawn on the screen.

The *RegularPolygon* class demonstrates encapsulation by combining all the variables and methods required for an instance together in one place. Many important variables are set when an instance is first created. Also, within the class constructor, *setCoords()* is run to calculate the points of the polygon.

|  |
| --- |
| class RegularPolygon {  int numberOfSides;  double interiorAngle;  double[] coords\_x;  double[] coords\_y;  double[] centerpoint;  double radius;  double sideLength;  RegularPolygon(int n, double[] cp, double r) {  numberOfSides = n;  centerpoint = cp;  radius = r;  interiorAngle = 360 / numberOfSides;  coords\_x = new double[numberOfSides];  coords\_y = new double[numberOfSides];  setCoords();  } |

Evidence of different classes can be seen in the existence of *RegularPolygon*, *JavaFXApplication1*, and *GraphicsContext*. Each of these serves a distinct role in the program. *RegularPolygon* is the class used to define what a polygon is. *JavaFXApplication1* is the main class in this scenario. *GraphicsContext* is used by JavaFX to draw art to the canvas.

The methods that I created for this program include *drawRegularPolygon(), drawCircle(), drawPolygonInscribed(), drawVenn(), drawPolygonStar()*, and *DrawRegularPolygonFractal()*. All of these methods perform a specific role in fabricating interesting geometric shapes. The names are very self-explanatory. *drawRegularPolygonFractal()* draws a smaller version of a polygon at every vertex of the original. It is a recursive function and therefore loops a certain number of times defined in the method call. As seen right, the effects of *drawPolygonStar()*, *drawRegularPolygonFractal()*, and *drawVenn()* can be quite interesting to view. All of these methods take identical input. They require *GraphicsContext, Color,* and *RegularPolygon* objects as well as a *lineWidth* integer. Using this information, the methods can adequately draw the shapes.

|  |
| --- |
| private void drawRegularPolygon(GraphicsContext gc, RegularPolygon polygon, Color color, int lineWidth) {  gc.setStroke(color);  gc.setLineWidth(lineWidth);  gc.strokePolygon(polygon.coords\_x, polygon.coords\_y, polygon.numberOfSides);  } |

The *main()* method consists of just the line *launch(args);* this is because the other methods in my program are for drawing on the canvas and therefore belong in the *start()* method.

I believe my project is uniquely set apart from my classmates because it uses trigonometric math to create geometric patterns rather than using gradients or images. I hope to use the principles I learned here in future projects, such as incorporation of math, and fractals.