

CS 152 Computer Programming Fundamentals

Project 3: Drawing with Points

Brooke Chenoweth

Spring 2024

In this assignment, you will write a program that uses a class I have provided for you to draw a picture or pattern of points.

1 The Display class

The `Display` class is a small little class. Its only function is to display points. The window keeps track of the last n points that were displayed and draws them as a trace of fading color.

- When creating the display window object, we will specify the number of trailing points and the size in pixels of each point.

For example, to create a `Display` object that will display a trace of fifty points, each five pixels wide, and assign it to a variable named “`window`”, you would write following line of code.

```
Display window = new Display(50, 5);
```

- To draw a point at a particular position, you will use the `placePoint` method. The coordinate system we use has $(0,0)$ at the upper left corner, with x increasing as we go to the right and y increasing as we go down.

For example, if we were using the `Display` variable above, we could draw a point at $(25,40)$ with the following line of code.

```
window.placePoint(25, 40);
```

- I’ve also provided you with two other methods to tell you how many pixels wide and how tall the display area is.

Again using the `Display` variable above, we can query the display size and assign the values to variables, like so.

```
int width = window.getWidth();  
int height = window.getHeight();
```

To use the `Display` class, you need to place `Display.java` in the `src` directory of your IntelliJ project along with the java file for the program that is using it. I have given you a couple example programs so you can try it out.

2 What you have to do

1. Create a class named `PointDrawings`.
2. Create a `Display` object in the `main` method.
You should only create one `Display` object for the entire program.
3. Call the `placePoint` method with a coordinate of `(200, 200)` and run your program. A point should be drawn at the center of the screen.

Try calling `placePoint` with other coordinates to see what happens. What happens if you put the method call inside a loop?

4. Now that you've started your program, change it to draw the following in succession.

- **Box** – Make the display window draw a point that moves around the screen to create a rectangle that is twice as tall as it is wide. (For example, if the box is 100 pixels wide, it should be 200 pixels tall.) This means that you will have to modify the indices of the point to move right, then down, then left, and then up. You will need to use several loops for this.

You may draw the box wherever you like and of whatever size you choose as long as the box has the required proportions, it fits on the window, and the point is shown moving around the perimeter of the rectangle in a continuous path.

- **Spiral** – I gave you an example of code that drew points on the circumference of a circle, where the angle was changing each iteration of the loop, but the radius was fixed. Consider what happens if you also change the radius. You'll draw a spiral.

Write a *single loop* that will draw a spiral shape on the same display that you used for the box. Draw the spiral from the outside in, moving in a clockwise direction. Make sure you stop spiraling when you get close enough to the center.

See the mathematical explanations in section 3, if you are feeling rusty on your trig functions and need a little help understanding the math in my circle demo code.

- **Something Fancy** – Come up with your own “fancy” version of a point drawing. The idea should be more complex than the movement on the circle – Can you come up with some nifty pattern?¹ Again, draw this third part on the same display as the first two parts.

Make sure that the loops for your box and spiral designs eventually terminate so the graders will be able to see all three parts of your program. The fancy third part can run forever if you like.

¹Past students have drawn smiley faces, stars, fractals, etc.

3 Circle trigonometry

Calculating the x and y coordinates for a point p on a circle as described in the figure below, is not too complicated if we use a little trigonometry.

Bascially, referring to the figure, we can calculate:

$$\begin{aligned}\Delta x &= r \cos \alpha \\ \Delta y &= r \sin \alpha\end{aligned}$$

The trigonometric methods in the java `Math` class, expects angles measured in radians, so if you are used to thinking of angles as degrees, you may want to convert those degrees to coordinates before applying the `sin` and `cos` methods. The conversion is simple: $\text{rads} = \frac{\pi \cdot \text{degs}}{180}$. Even better, the `Math` class has handy `toRadians` and `toDegrees` methods you can use to convert back and forth.

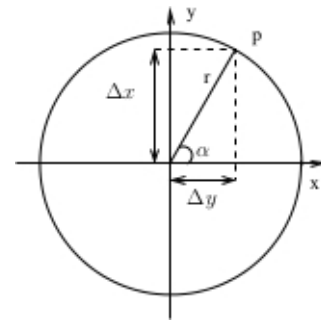


Figure 1: The measurements of a circle

4 Turning in your assignment

Submit your `PointDrawings.java` file to the Project 3 assignment in Canvas. Do not attach `.class` files or any other files. *Only submit one file*, a single program that draws all three required parts.