Recitation 5 - Loopy Loops

Objective

This Recitation will require you to write several functions in which loops will be the focus. So, let's get started!

At the end of this Recitation exerciseRecitation5, a studnet will be able to:

- Implemet for and while loops
- Translate shape equations to text patterns

Left Triangle

In the template below, Recitation5 has a method with the following signature:

```
public static String leftRightTriangle(int height)
```

that returns a string consisting of asterisks in a left-right triangle pattern with a height, **`height`**. When the method is invoked as follows:

```
Recitation5.leftRightTriangle(5);
```

the result is a string that is equivalent to the following:

```
*\n**\n***\n****\n*****\n
```

If printed,

```
System.out.println(Recitation5.leftRightTriangle(5));
```

The result is:

```
*
**
***
****
*****
```

Invoking the method with a different argument,

```
Recitation5.leftRightTriangle(3)
```

results in,

```
*\n**\n***\n
```

and if printed,

```
System.out.println(Recitation5.leftRightTriangle(3));
```

results in,

```
*
**
***
```

Right Triangle

In the template below, Recitation5 has a method with the following signature:

```
public static String rightRightTriangle(int height)
```

that returns a string consisting of asterisks in a right-right triangle pattern with a height, **`height`**. When the method is invoked as follows:

```
Recitation5.rightRightTriangle(5);
```

the result is a string that is equivalent to the following:

```
    *\n   **\n  ***\n ****\n*****\n
```

If printed,

```
System.out.println(Recitation5.rightRightTriangle(5));
```

The result is:

```
    *
   **
  ***
 ****
*****
```

Invoking the method with a different argument,

```
Recitation5.rightRightTriangle(3)
```

results in,

```
  *\n **\n***\n
```

and if printed,

```
System.out.println(Recitation5.rightRightTriangle(3));
```

results in

```
  *
 **
***
```

Let's Draw a Circle - Yay!

The template below has a static method with the following signature:

```
public static String circle(int radius)
```

That returns a string of asterisks representing a circle pattern with a radius, **`radius`**. When the method is invoked as follows:

```
Recitation5.circle(5)
```

the result is a string as follows,

```
   ******\n ********\n ********\n ********\n**********\n ********\n ********
```

If printed,

```
System.out.println(Recitation5.circle(5))
```

the result is,

```
  ******
 ********
 ********
 ********
**********
 ********
 ********
 ********
  ******
```

Invoking the method with a different argument,

```
Recitation5.circle(8)
```

the result is a string as follows,

```
"     ******\n   **********\n  ************\n  ************\n **********
***\n **************\n **************\n****************\n **************\
n **************\n **************\n  ************\n  ************\n   ***
*******\n     ******\n"
```

If printed,

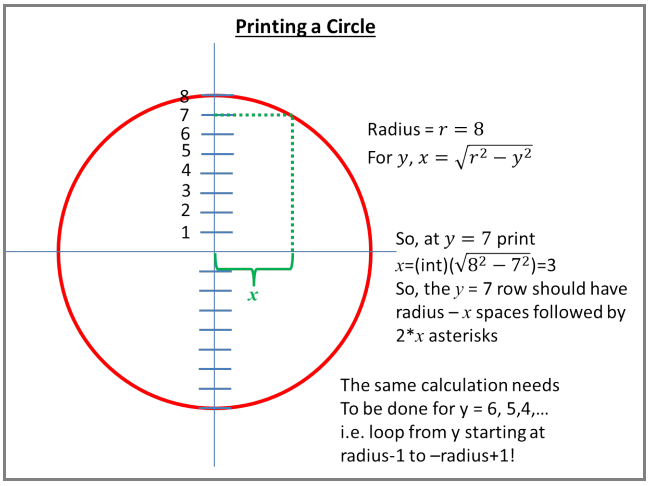
```
System.out.println(Recitation5.circle(8))
```

the result is,

```
      ******
    **********
  ************
  ************
 **************
 **************
 **************
****************
 **************
 **************
 **************
  ************
  ************
    **********
      ******
```

The above pictures look more of an oval than a circle. This is because of the aspect ratio of the text characters and the resolution of displaying it using characters. The height of a character is about twice the width. If you place a space after each asterisks, you will see that the image will look more like a circle.

An explanation of how to calculate the number of asterisks to print on each line is provided in the illustration below.



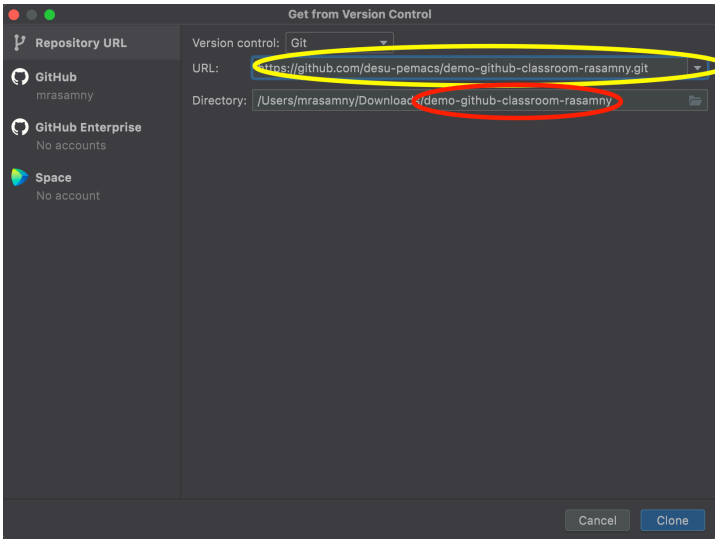
Note that you do not need to calculate x for $y = r$ or for $y = -r$ since for both these values, $x = 0$. So, you can use a for-loop that goes from **`radius-1 to -radius+1, inclusively or radius-1 to -radius, exclusively of -radius`**.

Cloning the Recitation

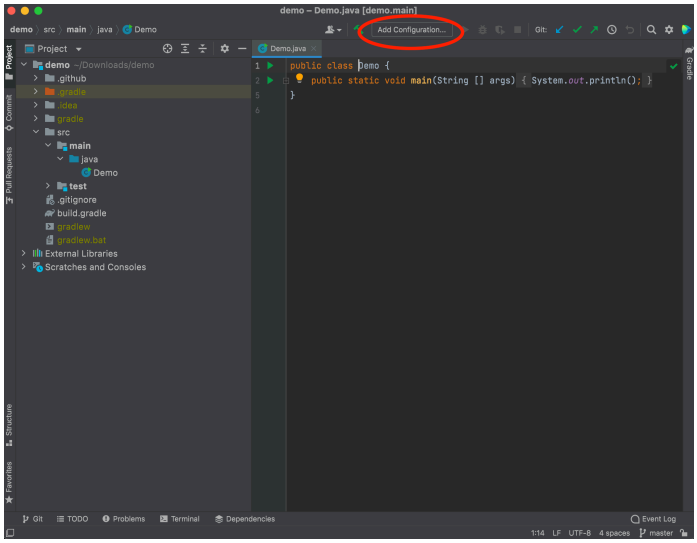
Click on the link in Blackboard or this link to accept the assignment on Github classroom. Once the repository is created, copy the link by clicking on the Clone button in your Github repository.

Do the following steps:

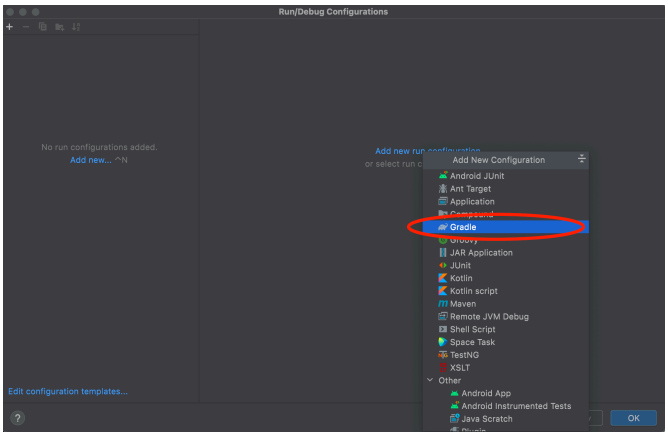
1. Click the **Get from VCS** button in IntelliJ, and paste the URL you copied from your Github repo in the textbox next to URL as shown in the picture. Make sure that the name of the repo also appears in the textbox under Directory.



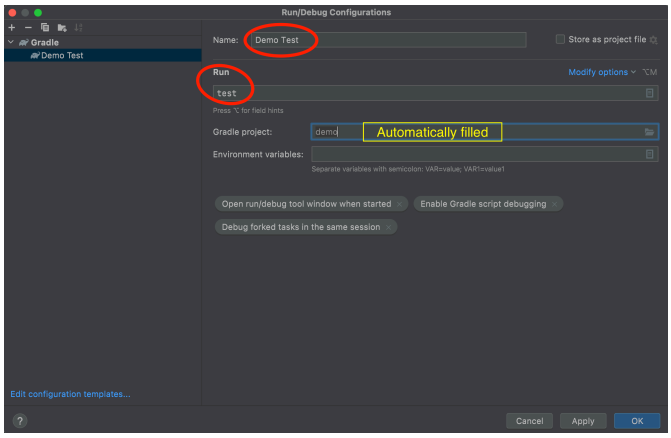
2. Press the Clone button. IntelliJ will create the folder for you and place the assignment code in the directory or folder with the name of the repo and your project should show on the IntelliJ window.
3. To set up the run configuration, click on the **Add Configuration** or **Edit Configuration** as shown in the image below.



4. select Gradle from the menu as shown in the image below.



5. Give the configuration a name, for example **Test**, and type test in the run textbox then click OK to complete the run configuration. NOTE: the image is based on the steps from the first recitation.



6. To run the tests, just press the **Green** play button as shown in the image below.