Homework 2

Before attempting this project, be sure you have completed all of the reading assignments, hands-on labs, discussions, and assignments to date.

Design a Java class named Polygon that contains:

- A private int data field named numSides that defines the number of sides of the polygon. The default value should be 4.
- A private double data field named sideLength that defines the length of each side. The default value should be 5.0.
- A private double data field named xCoord that defines the x-coordinate of the center of the polygon. The default value should be 0.0.
- A private double data field named yCoord that defines the y-coordinate of the center of the polygon. The default value should be 0.0.
- A private double data field named apothem that defines the apothem of the polygon. The default value should 5.0.
- A private double data filed named perimeter that defines the perimeter of the polygon. The default value should 20.0.
- A no argument constructor that creates a Polygon using the default number of sides, default side length, default x- and y-coordinates, and the default apothem.
- A constructor that creates a Polygon using a specified number of sides, side length, x- and y-coordinates and the apothem
- Getter and setter methods for all data fields
- A getArea() method that returns a double value representing the area of the polygon. (Note the area of a regular Polgon can be calculated from ½ * Apothem*Perimeter)
- A toString() method that displays the number of sides, side length, x-coordinate, y-coordinate and apothem values in String format

Be sure your code compiles. You should use the command prompt and not a GUI for data entry and display.

Write a Java test program, named TestPolygon, to create 5 different polygons representing the 5 test cases you just created. When creating the five polygons, create one using the no argument constructor. For the remaining four, feel free to use any number of sides, side length and x-, and y-coordinates and apothem that are not equal to the default values and not equal to each other. For each of the five polygons, call all of the methods and display the results. For example for a Polygon with 3 sides, side length of 2.0 and x-coordinate and y-coordinates of 1.0, and apothem of 1.0 the following test data may result:

Output

```
toString(): (numsides=3, sideLength=2.0, xcoord=1.0,ycoord=1.0, apothem=1.0)
getNumSides(): 3
getSideLength(): 2.0
getXCoord(): 1.0
getYCoord(): 1.0
getApothem():1.0
getPerimeter(): 6.0
```

```
getArea(): 3.0
```

Document your test cases in the form of table with columns indicating the input values, expected output, actual output and if the test case passed or failed. This table should contain 4 columns with appropriate labels and a row for each test case. An example template is shown below. Note that the actual output should be the actual results you receive when running your program and applying the input for the test record.

Keep in mind, for five Polygons, you will have five different output results. Also, note there is no requirement to actually draw a Polygon.

Example test cases:

Input	Expected Output	Actual Output	Pass?
Constructor: numsides=3 sideLength=2.0 xcoord=1.0 ycoord=1.0 apothem=1.0	** Output ** toString(): (numsides=3, sideLength=2.0, xcoord=1.0,ycoord=1.0, apothem=1.0) getNumSides(): 3 getSideLength(): 2.0 getXCoord(): 1.0 getYCoord(): 1.0 getApothem():1.0 getPerimeter(): 6.0 getArea(): 3.0	** Output ** toString(): (numsides=3, sideLength=2.0, xcoord=1.0,ycoord=1.0, apothem=1.0) getNumSides(): 3 getSideLength(): 2.0 getXCoord(): 1.0 getYCoord(): 1.0 getApothem():1.0 getPerimeter(): 6.0 getArea(): 3.0	**Yes**
Test case 2 here			
Test case 3 here			
Test case 4 here			
Test case 5 here			

The google recommended Java style guide, provided as link in the week 2 content, should be used to format and document your code. Specifically, the following style guide attributes should be addressed:

- Header comments include filename, author, date and brief purpose of the program.
- In-line comments used to describe major functionality of the code.
- Meaningful variable names and prompts applied.
- Class names are written in UpperCamelCase.
- Variable names are written in lowerCamelCase.
- Constant names are in written in All Capitals.
- Braces use K&R style**.**

Submission requirements

Deliverables include all Java files (.java) and a single word (or PDF) document. The Java files should be named appropriately for your applications. The word (or PDF) document should include screen captures showing the successful compiling and running of each of the test cases. Each screen capture should be properly labeled clearly indicated what the screen capture represents. The test cases table should be included in your word or PDF document and properly labeled as well.

Submit your files to the Homework 2 assignment area no later than the due date listed in your LEO classroom. You should include your name and HW2 in your word (or PDF) file submitted (e.g. firstnamelastnamehw2.docx or firstnamelastnamehw2.pdf)

Grading Rubric:

The following grading rubric will be used to determine your grade:

Attribute	**Meets**	**Does not meet**
Polygon Class	10 points Private int data field named numSides that defines the number of sides of the polygon. The default value should be 4. Private double data field named sideLength that defines the length of each side. The default value should be 5.0. Private double data field named xCoord that defines the x-coordinate of the center of the polygon. The default value should be 0.0. Private double data field named yCoord that defines the y-coordinate of the center of the polygon. The default value should be 0.0. Private double data field named apothem that defines the apothem of the polygon. The default value should 5.0. Private double data filed named perimeter that defines the	0 points Private int data field named numSides was not included. Private double data field named sideLength was not included Private double data field named xCoord was not included Private double data field named yCoord was not included Private double data field named apothem was not included. Private double data filed named perimeter was not included No argument constructor that creates a Polygon was not included Constructor that creates a Polygon using a specified number of sides, side length, x- and y-coordinates and the apothem was not included

	perimeter of the polygon. The default value should 20.0. No argument constructor that creates a Polygon using the default number of sides, default side length, default x- and y-coordinates, and the default apothem. Constructor that creates a Polygon using a specified number of sides, side length, x- and y-coordinates and the apothem Getter and setter methods for all data fields getArea() method that returns a double value representing the area of the polygon. (Note the area of a regular Polgon can be calculated from ½ * Apothem*Perimeter) toString() method that displays the number of sides, side length, x-coordinate, y-coordinate and apothem values in String format Command line was used for running the application.	Getter and setter methods for all data fields were not included getArea() method was not included toString() method was not included Command Line was not used for running application
Test Polygon Class	5 points TestPolygon used to create 5 different polygons representing the 5 test cases you just created. One test cases used the no argument constructor. An additional 4 test cases with varying input values.	0 points TestPolygon was not used to create 5 different polygons representing the 5 test cases you just created. One test cases was not used with the no argument constructor.

	For each of the five polygons, all of the methods were called and resulted displayed. Command Line was used for output.	An additional 4 test cases with varying input values were not used. For each of the five polygons, none of the methods were called Command Line was not used for output.
Test Cases	5 points A minimum of 5 test cases was used in the form of table with columns indicating the input values, expected output, actual output and if the test case passed or failed. The table should contains 4 columns with appropriate labels and a row for each test case. Test cases were included in the supporting word or PDF documentation.	0 points No test cases were provided.
Documentation and Style guide	5 points Screen captures were provided and labeled for compiling your code, and running each of your 5 test cases. Header comments include filename, author, date and brief purpose of the program. In-line comments used to describe major functionality of the code. Meaningful variable names and prompts applied. Class names are written in UpperCamelCase.	0 points No documentation included Java style guide was not used to prepare the Java code.

	Variable names are written in lowerCamelCase. Constant names are in written in All Capitals. Braces use K&R style**.**	