Practice Lab Assignment 2

For this Practice Lab Assignment, you will write programs to practice Java concepts taught in the class.

Instructions

- There are 5 questions in this assignment.
- Do not share your work with anyone.
- Discuss with TA in case of any further clarifications.

Due Date: 5th September 2024, Midnight.

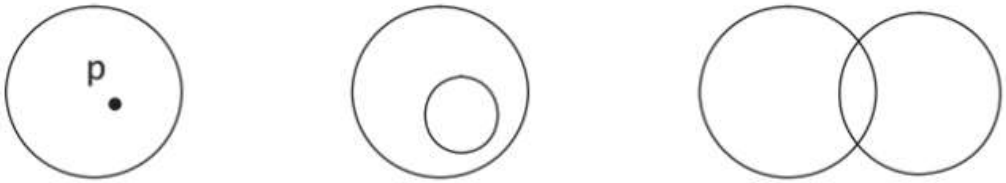
Submission Guidelines

1. You will submit (upload) this assignment in Blackboard. Email submissions will not be accepted.
2. Upload the .zip file containing .java files of all questions.
3. Name the .zip file as "John_Doe_2010110999" in case your name is "John Doe" and your roll number is "2010110999".

Questions

1. Define the Circle2D class that contains:

- Two double data fields named x and y that specify the center of the circle.
- A data field radius.
- A parameterless constructor that creates a default circle with (0, 0) for (x, y) and 1 for radius.
- A constructor that creates a circle with the specified x, y, and radius.
- A method getArea() that returns the area of the circle.
- A method getPerimeter() that returns the perimeter of the circle.
- A method contains(double x, double y) that returns true if the specified point (x, y) is inside this circle (see Figure a).
- A method contains(Circle2D circle) that returns true if the specified circle is inside this circle (see Figure b).
- A method overlaps(Circle2D circle) that returns true if the specified circle overlaps with this circle (see Figure c).



Implement the class and write a test program that creates a Circle2D object c1 (new Circle2D (2, 2, 5.5)), displays its area and perimeter, and displays the result of c1.contains(3, 3), c1.contains(new Circle2D(4, 5, 10.5)), and c1.overlaps(new Circle2D(3, 5, 2.3)).

2. Write a Program to design a class having static member function named show_count() which has the property of displaying the number of objects created of the class.

3. Implement a stack that supports push, pop, top, and retrieving the minimum element in constant time.

Implement the MinimumStack class with the below methods:

- **MinimumStack()** initializes the stack object.
- void **push(val)** pushes the element val onto the stack.
- void **pop()** removes the element on the top of the stack.
- int **peek()** returns the element at the top of the stack but does not remove.
- int **top()** gets the top element of the stack and also removes it from the stack.
- int **getMin()** retrieves the minimum element in the stack.

4. Use the above implementation of stack to ensure a given input has balanced brackets. The input string can contain elements from the given list of elements **" [,],{,},(,) "**

- The string {[()]} meets both criteria for being a balanced string.
- The string {[(])} is not balanced because the brackets enclosed by the matched pair { and } are not balanced: [(]).
- The string {{[[(())]]}} meets both criteria for being a balanced string.

5. Write a program to sort a list of values in ascending order. The user might Input a String array or an Integer array. Depending on the type of input, you need to call the **Void sort()** method. There are two variations of the method. Use any sorting method.

- **Void sort(int[] arr, int n)** - expects an integer array 'arr', and n denotes the size of the array
- **Void sort(int[] arr, int n, boolean reverse)** - expects an integer array 'arr', and n denotes the size of the array. It also has an additional `reverse` parameter which when set to **true** returns the array sorted in descending order.
- **Void sort(String[] arr, int n)** - expects a String array 'arr', and n denotes the size of the array. In this case, the array needs to be sorted lexicographically.

In this question, we are implementing three functions with the same name but different function parameters. What is this feature called in java and how is it related to OOP?