CSCI 2110 – Data Structures in Java
Lab exercise LabN1

- Deadline: Friday Jan 15th at 5pm
- Please first read the assignment check list available in the following link. If you do not follow the check list, you lose many points unfortunately. https://web.cs.dal.ca/~nourashr/teaching/cs2110/assignments/assignmentchecklist/
- If you have any question regarding the questions, please send an email to the instructor or TAs.
- Please submit your work on the Moodle portal:
  https://courses.cs.dal.ca/login/index.php
- You probably need to review lecture slides which are available in the following link. Feel free to use the codes of slides.
  https://web.cs.dal.ca/~nourashr/teaching/cs2110/lectures/
- We follow Dalhousie's policy of plagiarism.
  http://www.dal.ca/dept/university_secretariat/academic-integrity.html

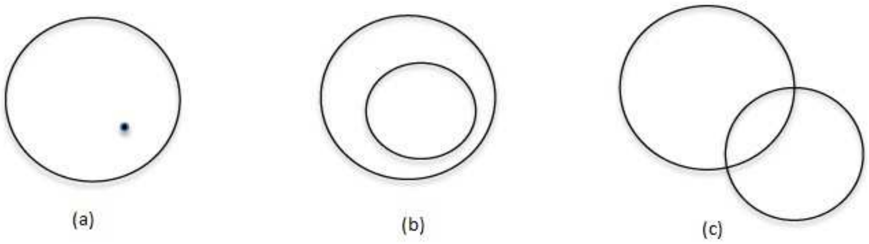
Q1) (25 points)

Define a Circle2D class that contains:

1. Two double data fields named x and y that specify the center of the circle.
2. A double data field radius.
3. Get and set methods for x, y, and radius. (5 points)
4. A no_arg (default) constructor that creates a default circle with (0, 0) for (x, y) and 1 for radius. (1 point)
5. A constructor that creates a circle with the specified x, y and radius of 1. (1 point)
6. A constructor that creates a circle with the specified x, y and radius. (1 point)
7. A method getArea() that returns the area of the circle. (1 point)
8. A method getPerimeter() that returns the perimeter of the circle. (1 points)
9. A method contains(double x, double y) that returns true if the specified point (x, y) is inside this circle (see Figure (a)). (5 points)
10. A method contains(Circle2D circle) that returns true if the specified circle is inside this circle (see Figure (b)). (5 points)

11. A method overlaps(Circle2D circle) that returns true if the specified circle overlaps with this circle object (see Figure (c)). (5 points)

Hint: circle2 with radius r2 is inside circle1 with radius r1 if the distance between the two centers <= |r1-r2| (that is, absolute difference), and circle2 overlaps circle1 if the distance between the two centers <= r1+r2.
Please test your implementation by showing two examples for items 7 to 11.



Q2) (15 points)

Write a Java class that extends the Progression class so that each value in the progression is the square root of the previous value. (Note that you can no longer represent each value with an integer.) You should include a default constructor that has 65536 as the first value and a parametric constructor that starts with a specified double number as the first value. Test your class by using a polymorphic object of the base class for default and non-default constructors. Four test cases to print sequences are enough.

Q3) (15 points)

Write a program that consists of three classes A, B, and C such that B extends A, and C extends B. Each class should define an instance variable named “x”. Describe a way for a method in C to access and set A’s variable of “x” to a given value. Show three different examples.

In the following question, assume that strings contain only numbers 0-9, lowercase letters a-z, and uppercase letters A-Z.

Q4) (15 points)

Write a method that inputs a text document and then outputs a bar-char plot of the frequency of each **alphabet** character that **appears** in that document. Two sample text files with minimum 100 words are enough to evaluate your method.

Q5) (30 points)

a) Use methods of java.lang.String class and write the method of removeDuplicate to remove duplicated character in a string. The definition of the class along with its methods are available in the following link. (http://docs.oracle.com/javase/7/docs/api/java/lang/String.html). (10 points)

   public static String removeDuplicates(String s) {….}

b) Write a method to decide if two strings are anagrams or not. Two strings are anagrams if they are made by rearranging their letters. For further information about Anagram, check the following link. (10 points)

   https://en.wikipedia.org/wiki/Anagram

   public static boolean anagram(String s, String t) {…}

c) Use methods of java.lang.String class and write the method of isSubstring which checks if one string is a substring of another. (5 points)

public static boolean isSubstring (String s1, String s2) { // returns true if s2 is a substring of s1 }

d) Given two strings, s1 and s2, write a method to check if s2 is a rotation of s1 using only one call to isSubstring (i.e., “waterbottle” is a rotation of “erbottlewat”). (5 points)

   public static boolean isRotation(String s1, String s2) {…}