**CS4013 Lab 1.**

In all of the following questions adopt an appropriate object-oriented design approach. Some of the following questions are adapted from “Introduction to Java” by Liang (9th edition)

1. Create a class in Java called MyPoint to represent a point in 2-dimensional space with x- and y-coordinates. The class should contain:
   1. The data fields x and y to represent the coordinates of the point;
   2. A no-arg constructor that creates a point (0,0);
   3. A constructor that creates a point with specified coordinates;
   4. Two accessor methods that return the values (as doubles) of the coordinates x and y;
   5. A method named distance that returns the distance from this point to another point of the MyPoint type; Note the distance between two point (x1, y1) and (x2,y2) is:
   6. A method named distance that returns the distance from this point to another point specified by its x and y coordinates.
2. **This question must be completed in pairs (P1 and P2).**

The Course class was defined in Chapter 10 of Liang. In this question you are asked to modify the Course class due to changing requirements of the client (customers/others who are currently using this class). However, you can only extend the contract of the class; you cannot modify the existing contract as some code which already uses Course.java will not be modified. You can access the source code of Course.java as follows:

* + Go to: <http://www.cs.armstrong.edu/liang/intro9e/examplesource.html>
  + On this page you can download all source code from the textbook (if you do this save on an external drive or on the cloud…you don’t want to have to repeat this step each time you need to access code from the book) or just access the individual file by following the link: <http://www.cs.armstrong.edu/liang/intro9e/html/Course.html>
* **Task 1**: **Work in pairs** to establish the current class contract by examining the implementation of Course.java. **Both P1 and P2 should do the following**. Add a javadoc comment(/\*\* … \*/) directly before the start of each method in Course.java. Now generate the javadocs (documentation) as follows:
  + **javadoc** –d ./docs \*.java This command generates the javadocs for all java files in the current directory and places these javadocs in the subdirectory called docs. Move to this directory and open index.html. Follow the links to see the documentation for your java file(s). The methods documented here form the class contract.
* **Task 2:** The following changes have been requested to Course.java:
* The client needs to be able to handle classes with more than 100 students. To do this **P1** should use and ArrayList to store the student names and **P2** should do this by creating a new larger array (maybe doubling its size) every time the existing array is full.
* **(P1 and P2)** Implement the dropStudent method which removes a student from the course. **Agree on the signature of the method which will now become part of the class contract.**
* **(P1 and P2)** Add a new method named clear that removes all students from the course. **Agree on the signature of the method which will now become part of the class contract.**
* **(P1 and P2)** Write a test program (a separate class and file which has the main method defined) that creates a course, add three students to the course, removes one and then displays the students in the course. You are now acting as the client and both test programs should be using the same class contract but note that the modified course classes have different implementations.

**If you have finished task 2 and your partner hasn’t, help your partner complete their task. (Do not write the code for them or take control of their keyboard). Share the completed implementation of Course.java with your partner.**

**For tasks 3 and 4 below P1 should work on (3) and P2 on (4)**

1. Create a class in Java to define the class Circle2D that contains:
2. Two double data fields named x and y that specify the centre of the circle;
3. A data field radius;
4. A no-arg constructor that creates a circle with (0,0) for the centre and 1 as the radius;
5. A constructor that creates a circle with specified x,y and radius;
6. A method getArea() that returns the area of the circle;
7. A method getPerimeter() that returns the perimeter of the circle;
8. A method contains(double x1, double x2) that returns true if the specified point is inside this circle and false otherwise.
9. A method contains(Circle2D circle) that returns true if the specified circle is inside this circle and false otherwise.
10. A method overlaps(Circle2D circle) that returns true if the specified circle overlaps with this circle and false otherwise.
11. Write a test program that creates two Circle2D objects, c1 and c2 and displays the area of each and displays the result of c1.contains(1,2); c1.contains(c2); c2.overlaps(new Circle2D(4,5,6));
12. Swap your test program with your partner. Execute your partner’s test program with your own implementation of the Circle2D class. Your program should work in the same way with the new implementation of Circle2D illustrating that the contract of each version of Circle2D is the same.
13. Create a Circle2D class as in (3) above but with the following changes:
    1. Store the centre of the Circle as a point (x,y) of type MyPoint.