LAB ASSIGNMENT #2

Due Date: Second class of Week 04 **Marks/Weightage:** 20/5%

Purpose: The purpose of this Lab Assignment is to:

- Practice the use of instance methods in Java classes
- Practice the use of static methods in Java classes

References: Read the course's text book **"Java How to program, 11th edition Early Objects"**, Chapters 06 and 07, the lecture notes/ppts, code examples and lab exercises. This material provides the necessary information that you need to complete the exercises.

Instructions: Be sure to read the following general instructions carefully:

This lab should be completed individually by all the students. You will have to demonstrate your solution in a scheduled lab session and submitting the assignment **through drop box link on e-Centennial**.

>> At the start, you must name your **Eclipse work space** according to the following rule:

FirstName_LastName_SectionNumber_COMP228_Labnumber

For Example: ***John_Smith_Sec006_COMP228_Lab02 (say if your section number is 006)***

>> And after that your **project name** should be as follows:

FirstName_LastName_SectionNumber _Labnumber

For Example: ***John_Smith_Sec006_Lab02***

>>Each exercise should be placed in a separate package named as firstname_last-name_*exercise1*, firstname_last-name_*exercise2* etc.

>> After you complete, exit eclipse and go to workspace folder, zip it up and you will get the following zip file.

FirstName_LastName_SectionNumber_COMP228_Labnumber.zip

Example: **John_Smith_Sec006_COMP228_Lab02.zip** *(if your section is 006..)*

>> Apply the naming conventions for variables, methods, classes, and packages:

- *variable names* start with a *lowercase* character for the first word and uppercase for every other word
- *classes* start with an *uppercase* character of every word
- **packages** use only *lowercase* characters
- *methods* start with a *lowercase* character for the first word and uppercase for every other word

Note: Late submissions are accepted until up to three days past due date with 25% deductions. After that no submission will be considered.

Exercise #1: [7 marks]

Write a Java application that simulates a test. The test contains **at least five** questions about first three lectures of this course. Each question should be a multiple-choice question with 4 options.

Design a **QuestionBank** class. Use programmer-defined methods to implement your solution. For example:

- create a method to simulate the questions – ***simulateQuestion***
- create a method to check the answer – ***checkAnswer***
- create a method to display a random message for the user – ***generateMessage***
- create a method to interact with the user - ***inputAnswer***

Use a loop to show all the questions.

For each question:

- If the user finds the right answer, display a random congratulatory message (“Excellent!”, ”Good!”, ”Keep up the good work!”, or “Nice work!”).
- If the user responds incorrectly, display an appropriate message and the correct answer (“No. Please try again”, “Wrong. Try once more”, “Don't give up!”, “No. Keep trying..”).
- Use random-number generation to choose a number from 1 to 4 that will be used to select an appropriate response to each answer.
- Use a switch statement to issue the responses, as in the following code:

```
switch ( randomObject.nextInt( 4 ) )
{
case 0:
return( "Very good!" );
……
}
```

At the end of the test display the number of correct and incorrect answers, and the percentage of the correct answers.

Your main class will simply create a QuestionBank object (in the driver class – **QuestionBankTest.java)** and start the test by calling **inputAnswer** method.

Exercise #2: **[7 marks]**

Design a **Lotto** class with one array instance variable to hold three random integer values (from 1 to 9). Include a constructor that randomly populates the array for a lotto object. Also, include a method in the class to return the array.

Use this class in the driver class (**LottoTest.java**) to simulate a simple lotto game in which the user chooses a number between 3 and 27. *The user runs the lotto up to 5 times (by creating an object of Lotto class each time and with that three random integer values will be stored in objects’s array instance variable) and each time the sum of lotto numbers (sum of three random integers values) is calculated*. *If the number chosen by the user matches the sum*, the user wins and the game ends. *If the number does not match the sum within five rolls*, the computer wins.

Exercise #3: **[6 marks]**

Write a Java class that implements a static method – **SortNumbers(int… numbers)** with variable number of arguments. The method should be called with different numbers of parameters and does arrange the numbers in descending order. Call the method within main method of the driver classand display the results.

Evaluation Rubric:

Functionality	
Correct implementation of classes as per requirements (instance variable declarations, constructors, getters and setters, methods etc.)	70%
Correct implementation of driver class (declaring and creating objects, calling their methods, interacting with user, displaying results)	20%
Comments, correct naming of variables, methods, classes, etc.	5%
Friendly input/output	5%
Total	100%