LAB #2 –Java Methods

Student: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Due Date: Week 4 (Oct 5, 11:59pm)

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 Lecture Notes #3, 4.

This material provides the necessary information you need to complete the exercises.

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 code through **through dropbox link on eCentennial**.

You must name your Eclipse project according to the following rule:

**YourFullName\_COMP228Labnumber**

Example: **JohSmith\_COMP228Lab2**

Each exercise should be placed in a separate package named *exercise1*, *exercise2*, etc.

Submit your assignment in a **zip file** that is named according to the following rule:

**YourLastName\_COMP228Labnumber.zip**

Example: **JohSmith\_COMP228Lab2.zip**

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

- *variable names* start with a *lowercase* character

- *classes* start with an *uppercase* character

- **packages** use only *lowercase* characters

- *methods* start with a *lowercase* character

**NOTE: SUBMIT A SINGLE PROJECT. USE PACKAGES TO DIVIDE YOUR WORK.**

**SUBMISSION OF INDIVIDUAL FILES WILL BE IGNORED.**

**Exercise 1:**

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 **Test** 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*

Display the questions using methods of ***JOptionPane*** class. 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 Test object and start the test by calling **inputAnswer** method.

(5 marks)

**Exercise 2:**

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 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 and each time the sum of lotto numbers 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.

Use methods of JOptionPane class to interact with the user.

(3 marks)

**Exercise 3:**

Write a Java class that implements a set of three overloaded static methods. The methods should have different set of parameters and perform similar functionalities. Call the methods within main method and display the results.

(2 marks)

**Evaluation:**

|  |  |
| --- | --- |
| **Functionality** |  |
| Correct implementation of classes (instance variable declarations, constructors, getter and setter methods, etc.) | 40% |
| Correct implementation of driver classes (declaring and creating objects, calling their methods, interacting with user, displaying results) | 40% |
| Comments, correct naming of variables, methods, classes, etc. | 5% |
| **Friendly input/output** | 15% |
| **Total** | 100% |