LAB ASSIGNMENT #1

Due Date: **Week 03** **Marks/Weightage: 10/5%**

Purpose: The purpose of this Lab Assignment is to:

- Practice the use of instance data members, constructors, methods in Java classes and objects

References: Read the course's text book **"Java How to program, 11th edition Early Objects"**, Chapter 03 and the lecture notes/ppts. 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 project **through drop box link on e-Centennial**.

You must name your Eclipse work space according to the following rule:
FirstName-LastName_SectionNumber_COMP228_Labnumber
For Example: **Joh-Smith_Sec001_COMP228_Lab01**

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

Submit your assignment in a **zip file** that is named according to the following rule:
FirstName-LastName_SectionNumber_COMP228_Labnumber.zip
Example: **Joh-Smith_Sec001_COMP228_Lab01.zip** *(if your section is 001..)*

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

Exercise #1: *[10 marks]*

Write a Java application using eclipse as IDE, that implements the following class(es) as per business requirements mentioned below:

Create a **BasePlusCommissionEmployee** class (***BasePlusCommissionEmployee.java***) that has the following instance variables:

- Employee ID, first name, last name, base salary, gross sales (amount in dollars) and commission rate. Define their data types appropriately.
- Define only getters for employee ID, first name, last name and base salary. Ensure the proper (no negative and null values) data values by implementing data validations.
- Use default value of 200.00 dollars for base salary for all the employees.

- Define getter and setter for gross sales and commission rate. Ensure their values should never be negative or zero.
- Commission rate should be between 0.1 and 1.0%. Set default value 0.1.
- Class should have defined two overloaded constructors:
  - One for initializing all the instance data members
  - Second for initializing employee ID, first name, base salary only.
- Define a public method - **double earnings()** which calculates employee's commission (commission rate * gross sales + base salary)
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class – **BasePlusCommissionEmployeeTest** (***BasePlusCommissionEmployeeTest.java***) which tests above class by at least creating two objects of the BasePlusCommissionEmployee class.

Exercise #2: *[10 marks]*

Write a Java application that implements the following class(es) as per business requirements mentioned below:

Create a **CheckingAccount** class (*CheckingAccount.java*) that has the following instance variables:

- Account number , customer name, account balance
- Define only getter for account number and customer name
- Define getter and setter for account balance. Balance should be positive and there should be minimum balance of 50.00 dollars all the time.
- Class should have defined a constructor:
  - For initializing all the instance data members
- Define one public method – public **double withdraw (double amount)** which is used for taking out money. With every withdrawal, there is transaction fee of 2.00 dollars.
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class – **CheckingAccountTest** (*CheckingAccountTest.java*) which tests above class by at least creating two objects of the CheckingAccount class with different set of data values.

Evaluation:

Functionality	
Correct implementation of classes (instance variable declarations, constructors, getter and setter methods etc.)	70%
Correct implementation of driver classes (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%