LAB ASSIGNMENT #1

Due Date: Second class of Week 03 **Marks/Weightage:** 20/5%

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, code examples and lab exercises. This material provides the necessary information that you need to complete the exercises.

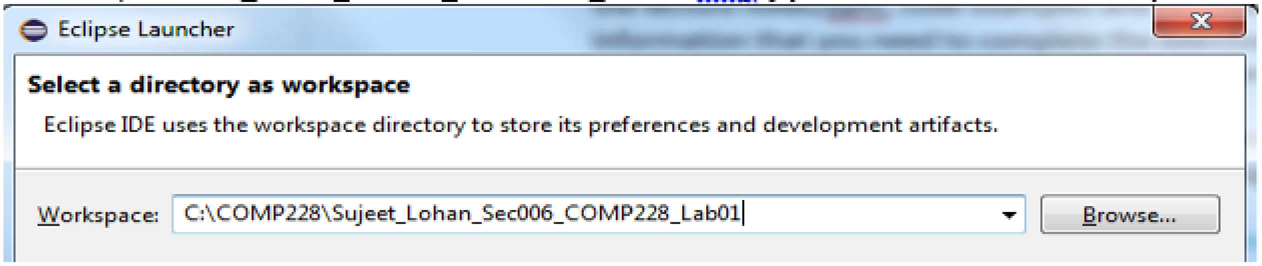
Instructions: Be sure to read the following general instructions carefully:

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_Lab01 (say if your section number is 006)***



Select a directory as workspace

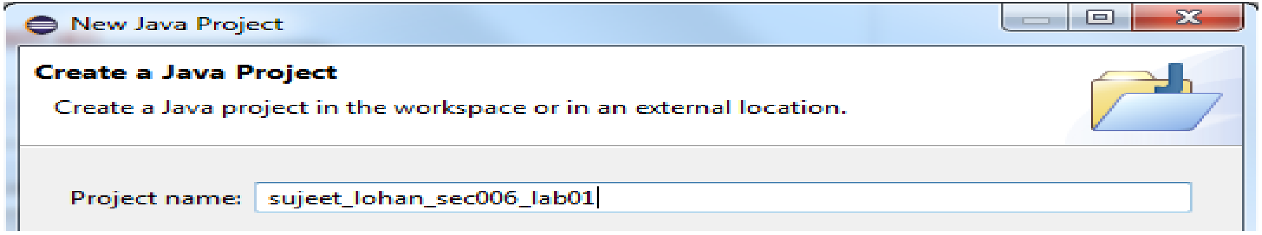
Eclipse IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\COMP228\Sujeet_Lohan_Sec006_COMP228_Lab01

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

FirstName_LastName_SectionNumber _Labnumber

For Example: ***John_Smith_Sec006_Lab01***



Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: sujeet_lohan_sec006_lab01

>>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_Lab01.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: *[5 marks]*

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

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

- Employee ID, first name, last name, gross sales (amount in dollars) and commission rate. Define their data types appropriately.
- Define only getters for employee ID, first name, last name. Ensure the proper (no negative and null values) data values entered by implementing data validations.
- Define getter and setter for gross sales and commission rate. Ensure the values for them 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/variables
  - Second for initializing employee ID, first name, last name
- Define a public method - **double earnings()** which calculates employee's commission (commission rate * gross sales/100)
- Define a public method – **String toString()** which is used to display the object's data

Create another **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 the values for them 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, last name, base salary only (if you want to set it more than 200.00 dollars)
- Define a public method - **double earnings()** which calculates employee's commission (commission rate * gross sales/100 + base salary)
- Define a public method – **String toString()** which is used to display the object's data

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

- Employee ID, first name, last name, total number of hours worked per week, and hourly rate (amount in dollars). Define their data types appropriately.
- Define only getters for employee ID, first name, last name. Ensure the proper (no negative and null values) data values by implementing data validations.

- Define getters and setters for number of hours worked per week and hourly rate. Ensure the values for them should never be negative or zero.
- Hourly rate should be minimum 14.00 dollars per hour.
- Class should have defined two overloaded constructors:
  - One for initializing all the instance data members/variables
  - Second for initializing employee ID, first name, last name
- Define a public method - **double earnings()** which calculates employee's earnings (number of hours worked per week * hourly rate)
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class **EmployeeTest** (***EmployeeTest.java***) which tests above classes by at least creating one object each of CommissionEmployee, BasePlusCommissionEmployee and HourlyEmployee classes.

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%