**Centennial College**

**COMP 228: Java Programming**

**LAB #1 – Java Class**

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

Due Date: Week 3

Purpose: The purpose of this Lab assignment is to:

* Practice the use Java classes with constructors, and getter and setter methods.

References: Learning materials for week 1 and 2, textbook, and other references (if any)

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 NEED TO SUBMIT THE FOLLOWING 2 DOCUMENTS IN THE DROPBOX TITLED LAB1:

1. THE FIRST ONE IS A WORD DOCUMENT. USE THIS DOCUMENT AND ADD SCREEN SHOTS OF THE RUNNING STATE OF EACH EXERCISE (If there are more than 1 exercise). DO NOT DELETE THE QUESTIONS. THE SCREEN SHOTS SHOULD FOLLOW EACH QUESTION AND COVER ALL THE ASPECTS/FUNCTIONALITIES OF EACH EXERCISE. AFTER THE SCREEN SHOTS PLEASE COPY THE CODE FROM THE CODE WINDOW AND PASTE THE COMPLETE CODE. DO NOT GIVE ME SCREEN SHOTS OF THE CODE. DO NOT ZIP THIS FILE AND KEEP IT SEPARATE FROM YOUR ZIPPED PROGAM FILE.
2. SUBMIT ALSO ONE ZIPPED PROJECT FILE THAT CONTAINS ALL THE EXERISES SEPARATELY INTO THE SAME DROP BOX.

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

**YourFullName\_COMP228Labnumber**

Example: **JohSmith\_COMP228Lab1**

Each exercise should be placed in a separate package(if there are more than 1 exercise) named *exercise1*, *exercise2*, etc.

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

**YourLastName\_COMP228Labnumber.zip**

Example: **JohSmith\_COMP228Lab1.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

**Exercise 1:**

Write a Java application that creates a Java console application to keep records of singers and displays the stored record. Follow the following instructions to develop the application:

Create a class named Singers with the following specifications:

* 5 instance variables that would store the following singer data (Use recommended variable naming conventions and appropriate data type for each instance variable):
  + Singer’s id
  + Singer’s name
  + Singer’s address
  + Date of birth
  + Number of albums published
* Several constructors that would allow you to construct Singer object with no arguments, 1 argument, 2 arguments, 3 arguments, 4 arguments, and 5 arguments.
* Create Setters and getters for all the instance variables of class Singer. Make sure to have several setters that would allow you to set the values of individual instance variables of the singer object. Also create one setter that would allow you to set all the values of the instance variables at once.
* Create several getters that would allow you to get the current individual values of each instance variables of the Singer object.
* Create the driver class that would create 1 Singer (singer1) object with the help of the no argument constructor.
* Display the default values of the instance variables of this object singer1.
* Now set all the values of singer1 with the help of the setter that sets all the values in one call.
* Display all the values of singer1 after setting all the values.
* Create another singer object (singer2) with the help of 5-argument constructor.
* Display all the values of singer 2.
* Modify some of the values of singer2 instance variables of your choice with the help of setters.
* Display the values of singer2 after the modifications.

**Evaluation:**

|  |  |
| --- | --- |
| **Functionality** |  |
| Correct implementation of classes (instance variable declarations, constructors, getter and setter methods, etc.) | 45% |
| Correct implementation of driver classes (declaring and creating objects, calling their methods, interacting with user, displaying results) | 45% |
| Friendly input/output and neat and clean organization of the submitted files | 10% |
| **Total** | 100% |