INTE 12213– Object Oriented Programming

Lab Exercises 02

You are given the following tasks. Perform all of them.

You have to use a text editor and the terminal in order to complete the below tasks and you are not permitted to use an IDE.

Prepare a report with the experiment, your observations on each task and conclusions. Submit the files by 11.59 pm on the lab session day to the CAL.

No marks will be given to any late submissions.

(Try to use Linux environment)

Lab02_Task01

Given the following class, called NumberHolder, write some code that creates an instance of the class, initializes its two member variables, and then displays the value of each member variable.

```
public class NumberHolder {

        public int anInt;

        public float aFloat;

}
```

Lab02_Task02

Write a very simple but complete class. The class represents a counter that counts 0, 1, 2, 3, 4,....

- The name of the class should be Counter.
- It has one private instance variable representing the value of the counter.
- It has two instance methods:
    - increment() adds one to the counter value, and
    - getValue() returns the current counter value.

Write a complete definition for the class, Counter.

Lab02_Task03

Modify class Account (Fig. 1) to provide a method called debit that withdraws money from an Account. Ensure that the debit amount does not exceed the Account's balance. If it does, the balance should be left unchanged and the method should print a message indicating "Debit amount exceeded account balance." Modify class AccountTest (Fig. 2) to test method debit.

Hints:

- Your output should appear as follows:

```
account1 balance: $50.00
account2 balance: $0.00

Enter withdrawal amount for account1: 25.67

subtracting 25.67 from account1 balance
account1 balance: $24.33
account2 balance: $0.00

Enter withdrawal amount for account2: 10.00

subtracting 10.00 from account2 balance
Debit amount exceeded account balance.
account1 balance: $24.33
account2 balance: $0.00
```

```java
public class Account
{
   private double balance; // instance variable that stores the balance

   // constructor
   public Account( double initialBalance )
   {
      // validate that initialBalance is greater than 0.0;
      // if it is not, balance is initialized to the default value 0.0
      if ( initialBalance > 0.0 )
         balance = initialBalance;
   } // end Account constructor

   // credit (add) an amount to the account
   public void credit( double amount )
   {
      balance = balance + amount; // add amount to balance
   } // end method credit

   /* write code to declare method debit. */

   // return the account balance
   public double getBalance()
   {
      return balance; // gives the value of balance to the calling method
   } // end method getBalance

} // end class Account
```

Fig. 1

```java
import java.util.Scanner;

public class AccountTest
{
   // main method begins execution of Java application
   public static void main( String args[] )
   {
      Account account1 = new Account( 50.00 ); // create Account object
      Account account2 = new Account( -7.53 ); // create Account object

      // display initial balance of each object
      System.out.printf( "account1 balance: $%.2f\n",
         account1.getBalance() );
      System.out.printf( "account2 balance: $%.2f\n\n",
         account2.getBalance() );
      // create Scanner to obtain input from command window
      Scanner input = new Scanner( System.in );
      double withdrawalAmount; // withdrawal amount read from user

      System.out.print( "Enter withdrawal amount for account1: " );
      withdrawalAmount = input.nextDouble(); // obtain user input
      System.out.printf( "\nsubtracting %.2f from account1 balance\n",
         withdrawalAmount );
      /* write code to withdraw money from account */

      // display balances
      System.out.printf( "account1 balance: $%.2f\n",
         account1.getBalance() );
      System.out.printf( "account2 balance: $%.2f\n\n",
         account2.getBalance() );

      System.out.print( "Enter withdrawal amount for account2: " );
      withdrawalAmount = input.nextDouble(); // obtain user input
      System.out.printf( "\nsubtracting %.2f from account2 balance\n",
         withdrawalAmount );
      /* write code to withdraw from account */

      // display balances
      System.out.printf( "account1 balance: $%.2f\n",
         account1.getBalance() );
      System.out.printf( "account2 balance: $%.2f\n",
         account2.getBalance() );
   } // end main

} // end class AccountTest
```

Fig. 2

Lab02_Task04

Modify class GradeBook (Fig. 3). Include a second String instance variable that represents the name of the course's instructor. Provide a set method to change the instructor's name and a get method to retrieve it. Modify the constructor to specify two parameter one for the course name and one for the instructor's name. Modify method displayMessage such that it first outputs the welcome message and course name, then outputs "This course is presented by: " followed by the instructor's name. Modify the test application (Fig. 4) to demonstrate the class's new capabilities.

```
public class GradeBook
{
   private String courseName; // course name for this GradeBook
   /* write code to declare a second String instance variable */

   // constructor initializes courseName with String supplied as argument
   public GradeBook( String name )
   {
      courseName = name; // initializes courseName
   } // end constructor

   // method to set the course name
   public void setCourseName( String name )
   {
      courseName = name; // store the course name
   } // end method setCourseName

   // method to retrieve the course name
   public String getCourseName()
   {
      return courseName;
   } // end method getCourseName

   /* write code to declare a get and a set method for the instructor's name */

   // display a welcome message to the GradeBook user
   public void displayMessage()
   {
      // this statement calls getCourseName to get the
      // name of the course this GradeBook represents
      System.out.printf( "Welcome to the grade book for\n%s!\n",
         getCourseName() );
      /* write code to output the instructor's name */
   } // end method displayMessage

} // end class GradeBook
```

Fig. 3

```java
public class GradeBookTest
{
   // main method begins program execution
   public static void main( String args[] )
   {
      // create GradeBook object
      GradeBook gradeBook1 = new GradeBook(
         "CS101 Introduction to Java Programming" );

      gradeBook1.displayMessage(); // display welcome message

      /* write code to change instructor's name and output changes */

   } // end main

} // end class GradeBookTest
```

Fig. 4

Hints:

- Your output should appear as follows:

```
Welcome to the grade book for
CS101 Introduction to Java Programming!
This course is presented by: Sam Smith

Changing instructor name to Judy Jones

Welcome to the grade book for
CS101 Introduction to Java Programming!
This course is presented by: Judy Jones
```

Lab02_Task05

Create a class called Employee that includes three pieces of information as instance variables—a first name (type String), a last name (type String) and a monthly salary (type double). Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0. Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects and display the yearly salary for each Employee. Then give each Employee a 10% raise and display each Employee's yearly salary again.

Lab03_Task06

Write an application that computes the total ticket sales of a concert. There are three types of seatings: A, B, and C. the program accepts the number of tickets sold and the price of a ticket for each of the three types of seats. The total sales are computed as

```
totalSales = numberOfA_Seats * pricePerA_Seat +

numberOfB_Seats * pricePerB_Seat +

numberOfC_Seats * pricePerC_Seat ;
```

Write this application using an instantiable SeatType class. An instance of the SeatType class keeps track of the ticket price for a given type of seat (A, B, or C) and the number of tickets sold for each seat type. (Hint: Create objects called seatA, seatB, and seatC to track the sales).