IS 380: Object-Oriented Programming
Spring 2017

Individual Assignment 3

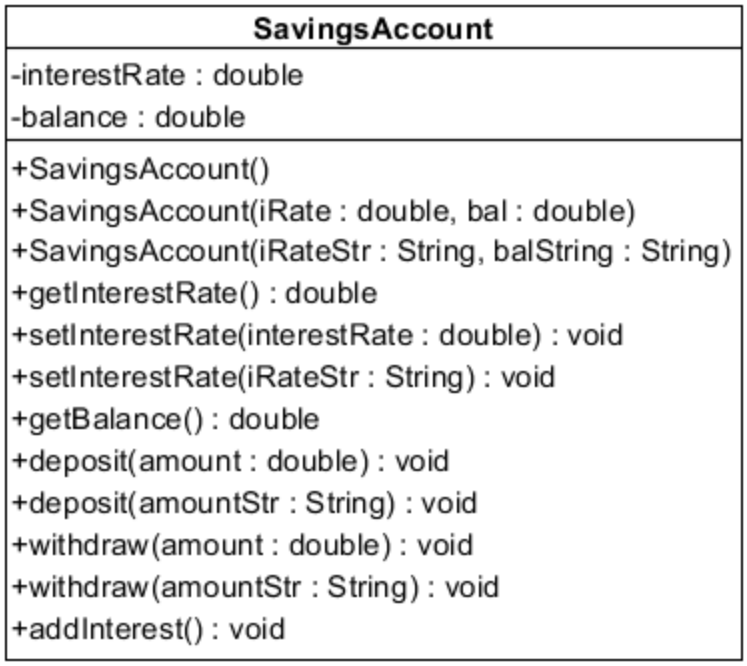
Due Date: **3:59 PM, March 8, 2017** (Submit via WebCampus).
Weights: 5% of total grades.

NOTE:

1. Please zip the **.java** files and upload the zip file to WebCampus for submission.
2. Please provide proper comments to document your code, including the following:
   a. Author's name;
   b. Purpose of the program;
   c. In-line comments for the statements.
   d. Method documentation using /** … */ and @param and @return tag.

QUESTIONS:

1. (40 %) Write a **SavingsAccount** class based on the specifications in the class diagram below.



(1) The class has two fields:
   (a) monthly interest rate: a double number
   (b) balance: a double number

(2) The class should have the following methods:
   (a) No-arg constructor: Set the interest rate and balance to 0.
   (b) Constructor: Accepts the double numbers interest rate and the amount of starting balance.
   If the input starting balance is less than zero, set the field **balance** to zero (0).
   If the input interest rate is less than 0, set the field **interestRate** to zero (0).
   If the input interest rate is greater than 0.01 (i.e., 1%), divide the number by 100 before setting the value to the field **interestRate**.
   (c) Constructor: Accepts two strings as the interest rate and the starting balance. (Note: Convert the String to numbers before setting the values to the fields.

(d) Get and set method for the interest rate, including the overloaded method with Strings as the parameters.
If the input interest rate is less than 0, set the field **interestRate** to zero (0).
If the input interest rate is greater than 0.01 (i.e., 1%), divide the number by 100 before setting the value to the field **interestRate**.

(e) **deposit** method: Add the amount of a deposit to the balance, including the overloaded method with String as the parameter.
If the input amount is less than 0, set the amount to zero(0).

(f) **withdraw** method: Subtract the amount of a withdrawal from the balance, including the overloaded method with String as the parameter.
If the input amount is less than 0, set the amount to zero(0).

(g) **addInterest** method: Adding the amount of monthly interest to the balance. To add the monthly interest to the balance, multiply the monthly interest rate by the balance, and add the result to the balance.

Grading criteria:

(1) Correctness (25 points):
  (a) The code can be compiled without any syntax error.
  (b) The fields, constructors, and the methods are correctly defined.
  (c) The program is properly documented using comments (/**…..*/ and //.)

(2) Technique used (15 points):
  (a) Fields and Methods are named properly, following the naming convention discussed in class.
  (b) Overloaded methods are implemented with different method signature.
  (c) Arithmetic operators are used.

2. (30 points) Write a program to test the **SavingsAccount** class.

(1) It should ask the user for the monthly interest rate, starting balance, and the number of months that have passed since the amount was established.

(2) A loop should then iterate once for each month, performing the following:
(a) Ask the user for the amount deposited into the account during the month. Use the class method to add this amount to the account balance.
(b) Ask the user for the amount withdrawn from the account during the month. Use the class method to subtract this amount to the account balance.
(c) Use the class method to add the monthly interest to the account balance.

(3) After the last iteration, the program should display the ending balance.

```
Please enter the starting balance: 1000
Please enter the monthly interest rate: 0.0003
How many months have passed by since the the account was established? 3
Please enter the withdraw amount for month 1: 500
Please enter the deposit amount for month 1: 300
The balance at the end of month 1 is $800.24
Please enter the withdraw amount for month 2: 200
Please enter the deposit amount for month 2: 400
The balance at the end of month 2 is $1000.54
Please enter the withdraw amount for month 3: 5000
Please enter the deposit amount for month 3: 8000
The balance at the end of month 3 is $4001.74
The ending balance is 4001.74
```

(4) the **DecimalFormat** class to format the resulting balance. The format of the balance should be something like **$585,655.80.**

Grading criteria:

(1) Correctness (15 points):
  (a) The code can be compiled without any syntax error.
  (b) The code can generate the requested results
  (c) The program is properly documented using comments (/**…..*/ and //.)

(2) Technique used (15 points):
  (a) Object of **SavingsAccount** is instantiated.
  (b) Either `while` loop, `do while` loop, or `for` loop is used to complete the program.
  (c) User input and program output are properly implemented.
  (d) **DecimalFormat** class is used to format the results.

3. (30%) Write a program that tosses four coins to get the balance.

(1) Use the **Coin** class you developed in the lab, and add a field to it as **faceValue**. (NOTE: the face value of a coin can be 1 cent, 5 cents, 10 cents, or 25 cents.)

(2) Add the get and set method for **faceValue** accordingly.

(3) Modify the constructor of the **Coin** class to assign the value for **faceValue**.

(4) Create a program to demonstrate this updated **Coin** class.
In this program, create four instances of the **Coin** class: one representing a quarter, one representing a dime, one representing a nickel, and one representing a penny.
After tossing the four coins, show the balance. The balance is calculated based on the face value of the coins landed heads-up.

Grading criteria:

(1) Correctness (15 points total)
  (a) The code can be compiled without any syntax error.
  (b) The **Coin** class is revised as requested.
  (c) The code can generate the requested results.
  (d) The program is documented using comments.
  (e) The program is properly documented using comments (/**…..*/ and //.).

(2) Technique used (15 points)
  (a) The program correctly uses the **Coin** class (i.e., instantiate the objects and use the methods).
  (b) The **faceValue** field of the **Coin** class is used.