Atılım University

CMPE 225 Object Oriented Programming
Fall, 2019-2020
Homework 3
T. ÜSTÜNKÖK
Due Date: January 10th, 2020, 23:59

Office Hours: 08/01/2020 – 11.00-16.00

The purpose of this assignment is to show you how the file and exception throwing/handling operations take place in C++.

Write a **BankAccount** class which has the following functions and member variables in it.

- **Member Functions:**
  - A constructor which initializes all member variables with the given parameters,
  - a virtual *save* function (returns nothing and takes nothing),
    - *Save* funtion creates a binary file which includes the concatenated values of all member variables of the current instance of the class. However, you should not create it by hand. All writing operations should be taken care by the write function of the *fstream* base class. As an example method, suppose you have a struct with member variables same as the *BankAccount* class. Then, you should call the write function such as:

      ```
      account_file.write((const char*) &var_struct, sizeof(VarStruct));
      ```

      When you check the file contents, you shouldn't be able to read it except for the raw strings.
  - a virtual *load* function with a parameter of a string (file name) and returns nothing,
    - *Load* function is almost similar to the save function. It takes a file name as parameter. Then, opens the file as a binary file, reads content of the file to the memory. You can use the same method in the save function. For example:

      ```
      account_file.read((char*) &var_struct, sizeof(VarStruct));
      ```

      Here, the *var_struct* variable contains all the loaded values for the *BankAccount* class.
  - a *draw* function which takes a float amount and returns a float,
    - *Draw*, if any, draws the specified amount of money from the account and returns the amount. If there is not enough money, it throws an *InsufficientDeposit* exception. You can define an empty class for this exception. There is nothing special about it.
  - an *apply_interest* function (returns nothing and takes nothing),
    - When called, this function applies the specified interest rate to the principal (the deposit). However, with different subclasses the effect of this function may vary.

    - a constant virtual *report* function (returns nothing and takes nothing).
        - The report function just prints a small report for the bank account which it is a member.
- **Member Variables:**
    - Integer *account_id*,
    - string *name*,
    - float *deposit*,
    - float *interest_rate* (protected)

Write 2 more classes *PersonalBankAccount* and *BusinessBankAccount*. Both inherits the BankAccount class. While initializing *PersonalBankAccount* class, change the interest rate to 18% and while initializing BusinessBankAccount class, change the interest rate to 6.5%.

For the *report* function, you should just add one more line to it to show the type of the bank account. You can find an example of it in the sample run.

In the main function, create 2 *PersonalBankAccount* and 1 *BusinessBankAccount*. Save them to their individual binary files. Then, load the second file to the first defined personal bank account and vice versa (in otherwords, swap the contents of them). Finally, draw excess amount of money from the *BusinessBankAccount* and write reports for all of the bank accounts.

The grading of the homework will be based on:

1. If you cheat, or high similarity percentage (above 85%), you will get 0.
2. In addition, if your code does not compile, your final grade will be over 50% of your total grade.
3. **Late submissions will not be graded.**

Good luck

```
 /\_/\
( o.o )
 > ^ <
```

Sample Run:

```
=========Before swapping accounts=========
TYPE: PERSONAL
==============
Account ID: 5421
Account Name: PersonalAcc1
Deposit: 8000
Interest Rate: 18%
TYPE: PERSONAL
==============
Account ID: 1234
Account Name: PersonalAcc2
Deposit: 5856
```

```
Interest Rate: 18%
=========After swapping accounts=========
TYPE: PERSONAL
==============
Account ID: 1234
Account Name: PersonalAcc2
Deposit: 5856
Interest Rate: 18%
TYPE: PERSONAL
==============
Account ID: 5421
Account Name: PersonalAcc1
Deposit: 8000
Interest Rate: 18%
InsufficientDeposit exception catched.
TYPE: BUSINESS
==============
Account ID: 5555
Account Name: BusinessAcc
Deposit: 70000
Interest Rate: 6.5%
```