<COMPUTER PROGRAMMING 2020 Assignment 1>

DUE DATE : 04/15/20 (Wed)

This assignment is to practice OOP programming with concepts in real life. We are going to implement a banking system with 3 types of bank accounts that controls all operations (sending money, putting money, taking money out). Please read the instructions below carefully.

class BankAccount

This is the abstract base class for all types of bank accounts.

It has the following data members

```
1. int acctnum;
```

- the account number

```
2. float bal;
```

- current balance of account

It has the following member functions

```
1. BankAccount(int num,float bal);
```

- parameterized constructor that takes two arguments and assigns them to variables acctnum and bal.

```
2. void deposit(float amount);
```

- increment the current balance by value of amount.

```
3. virtual int withdraw(float amount);
```

- decrement the current balance by value of amount.
- Then return 0;

```
4. int getAcctnum();
```

- getter function for acctnum variable

```
5. float getBalance();
```

- getter function for bal variable

```
6. virtual void print() = 0;
```

- pure virtual print function. You do not need to define it.

class Savings

This class is derived from BankAccount; it represents savings account and it has benefit of interest.

It has the following data members

```
1. float intrate;
```

- Interest rate for the savings account (annual)

It has the following member functions.

```
1. Savings(int num=0,float bal=0,float rate=3.5);
```

- Parameterized constructor that assigns following arguments to acctnum, bal, and, intrate.

```
2. void interest();
```

- It calculates interest for the month (not annual) based on current balance & interest rate and adds this value to current balance.
- Ex) current balance is 1200 and interest rate is 10% then your interest for the month is 10. Add that to 1200.

```
3. int withdraw(float amount);
```

- If current balance is less than or equal to the amount, do not execute withdrawal.
  - o Instead, print out message like below
  - o Cannot withdraw $(amount) on account #(acctnum) because the balance is low.
  - o Then return 0.
- Otherwise, decrement balance by the amount. Then return 1.

```
4. virtual void print();
```

- It needs to print a message like below.
- Savings Account: (account number)
  Balance: (current balance)
  Interest: (interest rate)%

class Checking

This class is derived from BankAccount; it represents checking accounts.

It has the following data members

```
1. float minimum;
```

- Minimum account balance to keep

```
2. float charge;
```

- Money charged if balance is lower than minimum at the time of withdrawal

It has the following member functions

```
1. Checking(int num=0,float bal=0,float min=1000,float chg=2);
```

- parameterized constructor that takes four arguments and assigns them to variables acctnum, bal, minimum, and charge.

```
2. int withdraw(float amount);
```

- If current balance is less than or equal to the amount, do not execute withdrawal.
  - Instead, print out a message like below.
  - Cannot withdraw $(amount) on account #(acctnum) because the balance is low.
  - Then return 0.
- Otherwise, check first if the current balance is below minimum.
  - If yes, decrement the balance by the value of amount + charge (person is paying charge for not keeping the balance above the minimum).
  - If it is not below minimum, just decrement by the amount only. At last, return 1.

```
3. virtual void print();
```

- It needs to print a message like below.
- Checking Account: (account number)
  Balance: (current balance)
  Minimum to Avoid Charges: (minimum balance)
  Charge per Check: (charge amount)

class InterestChecking

This class is derived from Checking; it represents checking account that has benefit of interest.

It has the following data members

```
1. float intrate;
```

- Interest rate for the interest checking account (annual)

```
2. float minint;
```

- Minimum balance that is required to get interest

```
3. float moncharge;
```

- Monthly charge when balance is below minint

It has the following member functions.

```
1. InterestChecking(int num=0,float bal=0,float cmin=1000,float imin=2500,float chg=2,float rate=2.5,float monchg=10);
```

- Parameterized constructor that assigns following arguments to acctnum, bal, minimum, minint, charge, intrate, and moncharge.

```
2. void interest();
```

- It calculates interest for the month (not annual) based on the current balance and interest rate and adds this value to current balance.
- If the balance is lower than the minimum balance for the account, do not add interest for the month, but deduct by monthly charge.
- Ex) current balance is 1200 and interest rate is 10% then your interest for the month is 10. Add that to 1200.

```
3. virtual void print();
```

- It needs to print a message like below.
- Interest Checking Account: (account number)
    Balance: (current balance)
    Minimum to Avoid Charges: (minimum balance)
    Charge per Check: (charge amount)
    Minimum balance for interest and No Monthly Fee: (minimum interest balance)
    Interest: (interest rate) %
    Monthly Fee: (monthly charge)

class BankSystem

This class represents a bank system that controls operations related to all kinds of accounts you have implemented.

It does not have any member variables but only the following member functions.

```
1. void transaction(BankAccount* from, BankAccount* to, float amount);
```

- It takes two bank accounts; withdraws money of the amount from first account and deposits that into second account.
- Please note, the sender of money needs to pay a transaction fee. It is 10.
- Sender loses amount + charge from the balance but the receiver only gets the amount (not including the fee.. it is for the bank).

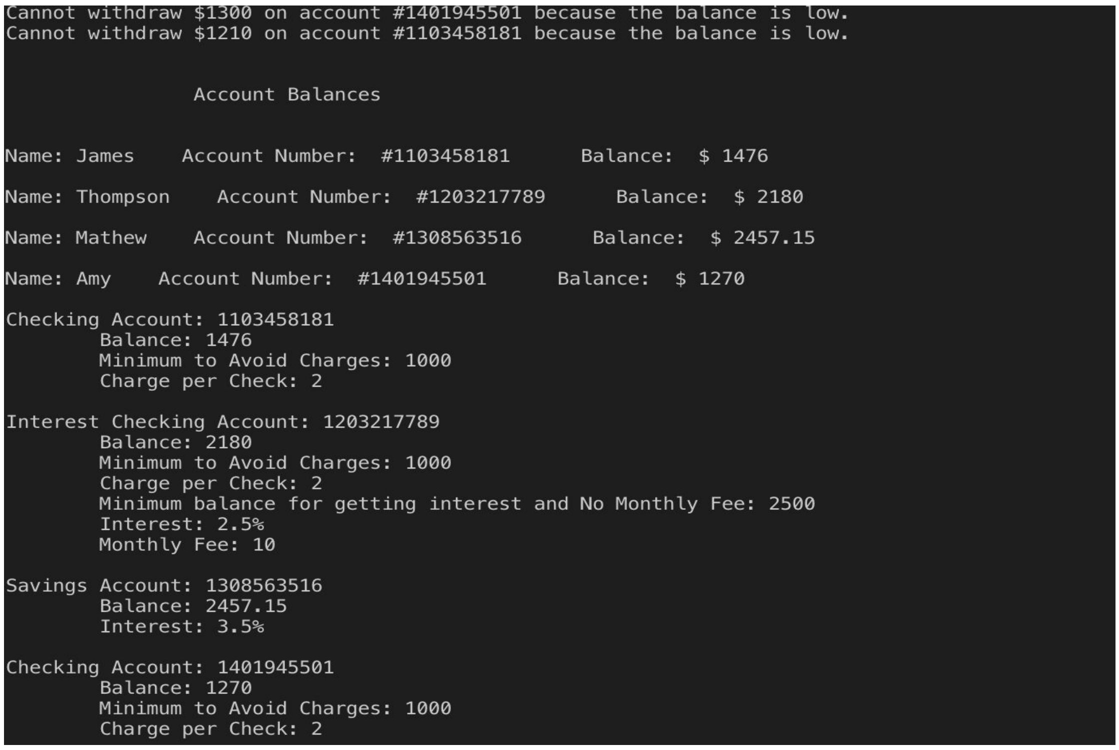
```
2. void deposit(BankAccount* b, float amount);
```

- It does not have a sender and receiver. It only takes one account and deposits the amount into its balance. Simple as it sounds.

```
3. void withdraw(BankAccount* b, float amount);
```

- It does not have a sender and receiver. It only takes one account and withdraws the amount from its balance. Simple as it sounds.

Expected output using main.cpp provided:



```
Cannot withdraw $1300 on account #1401945501 because the balance is low.
Cannot withdraw $1210 on account #1103458181 because the balance is low.


                Account Balances


Name: James    Account Number:  #1103458181      Balance:  $ 1476

Name: Thompson    Account Number:  #1203217789      Balance:  $ 2180

Name: Mathew    Account Number:  #1308563516      Balance:  $ 2457.15

Name: Amy    Account Number:  #1401945501      Balance:  $ 1270

Checking Account: 1103458181
        Balance: 1476
        Minimum to Avoid Charges: 1000
        Charge per Check: 2

Interest Checking Account: 1203217789
        Balance: 2180
        Minimum to Avoid Charges: 1000
        Charge per Check: 2
        Minimum balance for getting interest and No Monthly Fee: 2500
        Interest: 2.5%
        Monthly Fee: 10

Savings Account: 1308563516
        Balance: 2457.15
        Interest: 3.5%

Checking Account: 1401945501
        Balance: 1270
        Minimum to Avoid Charges: 1000
        Charge per Check: 2
```

IMPORTANT NOTE for the assignment

We are going to provide all header files for the classes and the main function file. Take a look at them carefully and do your best to get the **exact same output** as above.

For your convenience:

On visual studio code, Press ctrl-shift-c (or cmd-shift-c for mac) to open the terminal. Use cd command to move to the directory (or folder) that contains the files and use g++ to build and run the executable program. Good luck.

You must attend 4/7 and 4/14 Labs for further instructions and help.

GRADING CRITERIA FOR ASSIGNMENT 1

1. Correct logic and implementation for BankAccount (15pt)
2. Correct logic and implementation for Savings (20pt)
3. Correct logic and implementation for Checking (20pt)
4. Correct logic and implementation for InterestChecking (20pt)
5. Correct logic and implementation for BankSystem (15pt)
6. Printing same output as given (including number of spaces and so on) (10pt)

TOTAL: 100pt

SUBMISSION:

Submit all your files with provided headers and main.

Zip them and make the zip file name “cp2020hw1_(your student id number)_(your name in english).zip”