CISC2000: Computer Science II Lab 6

1. Introduction

This is a new lab to help you learn how to use inheritance and multiple files for separate compilation to create a program. The program is a simple ATM. (Automated Transaction Machine). It starts up and asks to update current info for the two accounts. Don't change the input or output wording as it matches the I/O test cases. You will be able to test your input, output, deposit, withdraw methods in your CheckingAccount class using this program.

2. Getting Starter Code

The starter code is presented in either a zip file called ATM2000.zip or as separate source files listed below. If you want to take the single zip file, you will need to unzip it as shown below. It will automatically be placed in a directory called ATM2000.

```
[harazduk@storm cs2000]$ unzip ATM2000.zip
Archive:  ATM2000.zip
   creating: ATM2000/
  inflating: ATM2000/SavingAccount.h
  inflating: ATM2000/SavingAccount.cpp
  inflating: ATM2000/makefile
  inflating: ATM2000/CheckingAccount.h
  inflating: ATM2000/CheckingAccount.cpp
  inflating: ATM2000/BankAccount.h
  inflating: ATM2000/BankAccount.cpp
  inflating: ATM2000/atm2000.cpp
[harazduk@storm cs2000]$ ls ATM2000
atm2000.cpp  BankAccount.cpp  BankAccount.h  CheckingAccount.cpp
CheckingAccount.h  makefile  SavingAccount.cpp  SavingAccount.h
[harazduk@storm cs2000]$
```

3. Starter Code

The main program is in a file called `atm2000.cpp`. Some of the code for the main is provided for you in this file. Some you will have to provide.
There are comments in the file explaining what **`// To Do:`**
The starter code comes with two fully implemented classes called: `BankAccount` and `SavingAccount`. They should be familiar after the recent lecture. I reimplemented the work in the BankAccount 10-06 version which uses a double for the balance. I thought it would be easier for you to manage. If you remember, I said that the BankAccount class should automatically have deposit and withdraw methods rather than update. I made those changes. This way, every BankAccount can deposit and withdreaw but not all BankAccounts have interest.

The starter code for the classes and program are in separate files:
`atm2000.cpp BankAccount.h BankAccount.cpp SavingAccount.h SavingAccount.cpp`

You are also given two files for the `CheckingAccount` class:
`CheckingAccount.h CheckingAccount.cpp.`

4. Compiling

There are two ways to compile this program:

```
g++ atm2000.cpp BankAccount.cpp SavingAccount.cpp CheckingAccount.cpp -o atm2000
```

or

```
g++ -c BankAccount.cpp

g++ -c SavingAccount.cpp

...

g++ atm2000.cpp BankAccount.o SavingAccount.o CheckingAccount.cpp -o atm2000
```

5. Finish CheckingAccount.h, CheckingAccount.cpp

Your job is to complete the CheckingAccount class as a derived class of BankAccount and fill in the program details in atm2000.cpp. Put the declaration of the class in `CheckingAccount.h` and the implementation in `CheckingAccount.cpp`. Make sure to follow the comments that start with **`// To Do:`**

- Complete three constructors as we did with the SavingAccount class. The check fee should be handled exactly like the interest_rate. Remember that I used 10-06, for this assignment so we saved the interest_rate as a percentage. Here the check fee is an amount of dollars and cents represented as a double.
- Complete the accessor function for the new data member check fee.
- Complete the `output` function to include the check fee. Use `BankAccount::output(outs)` first.
- Complete the `input` function to include the check fee. Use `BankAccount::input(ins)` first.
- Complete the `check` function. It should behave like `withdraw` in BankAccount except that the check fee should also be subtracted.

DON'T FORGET TO PROTECT THE CLASS Declaration with either `#ifndef` or `#pragma once`
DON'T FORGET TO INCLUDE THE HEADER FILES WHERE NECESSARY.

6. Finish atm2000.cpp

The other part of your job is to complete the program in atm2000.cpp. Follow the **// To Do** comments

```
#include <iostream>

#include <string>

// To Do: Include the approproiate header files
// Utility functions
int main()
{
 // To Do: Declare a SavingAccount and a CheckingAccount object

 char transact, fromAcct;
 double amount;
 cout << "Update the account info for Checking (balanace and fee): ";

 // To Do: Call your input function for the CheckingAccount
 cout << "Update the account info for Saving (balanace and rate): ";

 // To Do: Call your input function for the SavingAccount
 cout <<
"\n\n*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-\n";
 cout << "***** Welcome to the ATM 2000!!\n";
 cout << "\n*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-\n";

 cout << "Looks like you accrued some interest in your Savings.\n";
 // To Do: Call your update function for the SavingAccount
 do {

  getTransaction(transact, amount, fromAcct);

  cout <<
"\n*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-\n";

  // To Do: take apart the amount into dollars and cents


  // To Do: Use the values returned from getTransaction to implement
the ATM

  // if transact contains
  // 'B' - call your output function on fromAcct ('C'=checking,
'S'=saving)
  // 'D' - call your deposit function on fromAcct ('C'=checking,
'S'=saving)
```

```
  // 'W' - call your withdraw function on fromAcct ('C'=checking,
'S'=saving)
  // 'T' - call your withdraw function on fromAcct ('C'=checking,
'S'=saving)
  //     - then call your deposit function on the other.
  // 'C' - call your check function on checking

 } while (transact != 'Q');

 cout << "Come Again! Bye!";

 return 0;

}
```

7. Sample Output

```
Update the account info for Checking (balanace and fee): 589.50 1.50
Update the account info for Saving (balanace and rate): 2655.70 2.00

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
***** Welcome to the ATM 2000!!

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-

Looks like you accrued some interest in your Savings.
What would you like to do today?
B)alance, D)eposit, W)ithdraw, C)heck, T)ransfer, Q)uit?
D
What amount would you like to Deposit today?
200.00
From which account? C)hecking S)aving?
C

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
What would you like to do today?
B)alance, D)eposit, W)ithdraw, C)heck, T)ransfer, Q)uit?
B
From which account? C)hecking S)aving?
C

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
Account Balance: $789.50
Check fee: $1.50
What would you like to do today?
B)alance, D)eposit, W)ithdraw, C)heck, T)ransfer, Q)uit?
W
```

```
What amount would you like to Withdraw today?
150.00
From which account? C)hecking S)aving?
C

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
What would you like to do today?
B)alance, D)eposit, W)ithdraw, C)heck, T)ransfer, Q)uit?
B
From which account? C)hecking S)aving?
C

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
Account Balance: $639.50
Check fee: $1.50

What would you like to do today?
B)alance, D)eposit, W)ithdraw, C)heck, T)ransfer, Q)uit?
Q

*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-*+-
Come Again! Bye!
```

- CheckingAccount Check Test

```
500.50 1.75
2000.00 2.45
C
235.55
B
C
C
57.35
B
C
Q
```

8. Submitting to Autograder

When submitting to Autograder, you should only submit the changed files. That includes `atm2000.cpp CheckingAccount.cpp` and `CheckingAccount.h`. This can be done by using the gray browse button in the middle of the submit window and clicking the files while holding down the Control (Ctrl) key. It might be different on a MAC, but I will update the instructions shortly.