CS 485: Assignment 5 45 points: Execution: 18 pts, Coding Standards: 9 pts, Design 18 points

Bank Accounts DUE: UML Diagram, **Wednesday** Apr 5, 3:30pm **(Hard Copy)**
DUE: Completed Solution, Monday, Apr 17, 3:30 pm

Goals for this assignment

1. Fix your 03_Bank
2. You are **not** required to match the design presented on March 3.
3. Use a **Singleton** to provide a Currency Conversion table
4. Add an Account Container class backed by an STL::map. Use the Account ID as the key.
5. Provide an interface to use the **visitor** pattern to your Account Container
6. Practice new C++ syntax and idioms
7. **Review your CS 250 C++ Notes**
8. This is not an SDL application

► You need to add a class that provides a Currency Conversion table. Any mathematical operation or comparison operation between two Money objects must perform currency conversion automatically. Use the currency of the object handling the method call as the currency of the result of the operation. The currency table will be read from the file CurrencyConversions.txt

► You must add an Account Container backed by an STL::map. Use this new map backed container in your code.

► Provide an interface to use the Visitor pattern with the Account Containers (both the array backed and map backed containers). **Use the visitor pattern to backup all accounts to files using the new B command.**

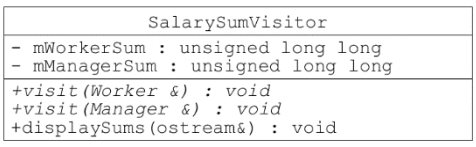
► All initial balances, fees, minimum balances, and interest tiers for a particular account are no longer required to use the same currency, however, these currencies will always be such that they are convertable.

► A currency mismatch exception will only occur if there is no conversion factor specified between the two currency types used in an operation. For this project, any command that generates a currency mismatch exception should produce no effect on any account, not display any message to the screen, and must not crash the program.

Your software will read the accounts from a file (Accounts.txt). Your software will also read the commands from a file (Commands.txt) and process the commands in order. These files are specified below.

You must produce a UML diagram of your design. This must be the design for the entire Bank system.

You must specify datatypes for member variables, function parameters, and return values as shown at right.



Highlight the new classes or edited classes by placing `lw=5` at the end of the Properties description for the class. The diagram must show inheritance, aggregation, and composition. You may choose to show other connections. **I strongly recommend** you bring your UML diagram to me for a review well before it is due. Use UMLet (http://umlet.com/) to produce your diagram. Include the XML file output by UMLet in your Visual Studio solution (just save the XML file into the **Visual Studio project directory**.)

Name your Visual Studio solution: CS485_Bank*_punetid*. Name your Visual Studio project: 04_Bank. Copy all of your code from 03_Bank to 04_Bank as a starting point. Follow the same procedure as last time to make this copy.
You must use Git and put your VS Solution into the private Bank*_githubname* repository created for you on the CS485s17 GitHub organization.

Follow the coding standards posted on the class website. Be sure to use mem_debug or VLD to check for memory leaks.

For this assignment, you will not receive any bad data in the files.

You must always display the accounts in ascending sorted order using the Account ID.

Data File Format

CurrencyConversions.txt

```
USD GBP 0.76
USD EUR 0.92
YEN EUR 0.69
YEN GBP 0.58
EUR GBP 0.84
```

The line USD GBP 0.76 means that 1 USD is worth 0.76 GBP.

The line EUR GBP 0.84 implies the line
GBP EUR 1.19

Not all pairs of currency will necessarily have a conversion factor.

The Accounts.txt remain the same.

A new command is added:

B checkingBackup_1.txt savingsBackup_1.txt

This command will cause all checking accounts to be written to the first file and all savings accounts to be written to the second file. This must be implemented by one vistor that visits each account once and writes that account to the correct file. Write the account to the file in exactly the same format you write the account to the screen. No dashed lines are written to the file.

Example Files

Use **CurrencyConversions.txt** from above.

Accounts.txt

```
S 1 USD 10000 F 0.01 USD 100 USD 100
C 4 USD 90000 F 0.01 USD 100 USD 100
S 3 GBP 10000 F 0.01 GBP 100 USD 100
```

Commands.txt

```
P
W 1 GBP 100
D 3 USD 100
D 1 YEN 1000
P
B c.txt s.txt
M
P
W 1 USD 9900
P
M
P
D 1 USD 102
P
M
P
P
M
P
D 1 GBP 100
W 3 GBP 100
P
W 4 YEN 1000
P
```

c.txt

```
4, USD900.00, F 1.00%, USD1.00, USD1.00
```

s.txt

```
1, USD98.69, F 1.00%, USD1.00, USD1.00
3, GBP100.76, F 1.00%, GBP1.00, USD1.00
```

Expected Output

```
-------------
1, USD100.00, F 1.00%, USD1.00, USD1.00
3, GBP100.00, F 1.00%, GBP1.00, USD1.00
4, USD900.00, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD98.69, F 1.00%, USD1.00, USD1.00
3, GBP100.76, F 1.00%, GBP1.00, USD1.00
4, USD900.00, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD99.67, F 1.00%, USD1.00, USD1.00
3, GBP101.76, F 1.00%, GBP1.00, USD1.00
4, USD909.00, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD0.67, F 1.00%, USD1.00, USD1.00
3, GBP101.76, F 1.00%, GBP1.00, USD1.00
4, USD909.00, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD-0.33, F 1.00%, USD1.00, USD1.00
3, GBP102.77, F 1.00%, GBP1.00, USD1.00
4, USD918.09, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD0.69, F 1.00%, USD1.00, USD1.00
3, GBP102.77, F 1.00%, GBP1.00, USD1.00
4, USD918.09, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD-0.31, F 1.00%, USD1.00, USD1.00
3, GBP103.79, F 1.00%, GBP1.00, USD1.00
4, USD927.27, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD-0.31, F 1.00%, USD1.00, USD1.00
3, GBP103.79, F 1.00%, GBP1.00, USD1.00
4, USD927.27, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD-1.31, F 1.00%, USD1.00, USD1.00
3, GBP104.82, F 1.00%, GBP1.00, USD1.00
4, USD936.54, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD0.00, F 1.00%, USD1.00, USD1.00
3, GBP103.82, F 1.00%, GBP1.00, USD1.00
4, USD936.54, F 1.00%, USD1.00, USD1.00
-------------
-------------
1, USD0.00, F 1.00%, USD1.00, USD1.00
3, GBP103.82, F 1.00%, GBP1.00, USD1.00
4, USD936.54, F 1.00%, USD1.00, USD1.00
-------------
```