CUS1156 Extra Credit Project

Recall the question on the exam about adding a list of transactions to a **BankAccount** class. In this project, you are going to implement that idea and extend it quite a bit. You will end up with a menu based system that allows users to withdraw, make deposits and print statements for a group of bank accounts that record transactions.

You can download a BankAccount.java class from Blackboard. Change it to add the transaction logging feature as described in the exam question. Each **BankAccount** should maintain a list of transactions. When a withdrawal or a deposit occurs, a transaction should be added to the list.

You will need a **Transaction** class which contains the amount involved in the transaction, the date/time of the transaction, and the ending balance. The **Transaction** class should have setters and getters, and a **toString** method. The augmented **BankAccount** class should maintain an array of **Transactions**. The **makeStatement** method should be changed so that it prints out the transactions.

Add the following subclasses to **Transaction : WithdrawTransaction**, **DepositTransaction**, **TransferTransaction**, and **ErrorTransaction**. The **ErrorTransaction** should be created and added to the list of transactions whenever there is an error. In your case, the two errors are withdrawals with insufficient funds, and trying to transfer money when the accounts are owned by different people. The second error condition was an extra credit feature in the last assignment, but for this one, you need to have it.

Your subclasses need constructors with the following interfaces :
public DepositTransaction(**double** amt, **double** endBal)
public WithdrawTransaction(**double** amt, **double** endBal)
public TransferTransaction(String fromAcctNumber, String toAcctNumber, **double** amt, **double** balance**)**
public ErrorTransaction(String acctNumber, String errorDescription, **double** amt, **double** balance)

The error description which is passed to the **ErrorTransaction** should contain messages such as "`ERR2556B - insufficient funds`". These error descriptions will be printed out when the transactions are printed.

You also will need toString methods in your various transaction classes. The toString method for **DepositTransaction** should print something like this :

```
Deposit : Mon Nov 01 10:55:40 EDT 2010 amount : 200.0 End
balance : 125.0
```

The toString method for **WithdrawTransaction** should print something like this :

```
Withdrawal : Mon Nov 01 10:55:40 EDT 2010 amount : -245.0 End
balance : 600.67000000
```

Notice that the amount is printed with as a negative number.

The toString method for **TransferTransaction** should print something like this :

```
Transfer : Mon Nov 01 10:55:40 EDT 2010 amount : 200.0 From
account 18234865 To account 1823454
```

Note that you will also see the deposit and withdrawal transactions corresponding to the transfer as well.

And finally the toString method for **ErrorTransaction** should print something like this :

```
Error : Mon Nov 01 10:55:40 EDT 2010 ERR2312A - not authorized
to make this transfer amount : 100.0 account 3674542
```

To test this set of classes, you can download the file BankAccountTester.java from Blackboard.

Once you have this working, you will create a **Bank** class. The **Bank** class will keep a list of **BankAccount** objects. It should allow a **BankAccount** object to be added to the list, and it should have a method that returns a **BankAccount** object based on the account number.

Finally, you should create a class that represents the user interface. It should read bank account information from a file which looks like this (you should add more accounts)

```
0.0,2223,Jones,Fred
100.56,3334,Lee,Karen
300.45,3456,Spacca,Mary
56.00,8978,Martin,Kelly
```

A **Bank** object should be created and loaded with this information. Then, a menu should be displayed that allows users to withdraw money, deposit money, print their statement or quit. Here is sample output

```
Options
d: deposit
w: withdraw
p: print statement
q: quit
d
Enter account number
3334
Enter amount to deposit
200
Account : 3334Your current balance is 300.56
Options
d: deposit
w: withdraw
p: print statement
q: quit
w
Enter account number
3456
Enter amount to withdraw
50
Account : 3456Your current balance is 250.45
Options
d: deposit
w: withdraw
p: print statement
q: quit
p
Enter account number
3334
Bank Account 3334 Statement Date :Tue Mar 26 17:57:57 EDT 2013 Current balance :
300.56
Deposit : Tue Mar 26 17:57:29 EDT 2013 amount : 200.0 End balance : 300.56

Options
d: deposit
w: withdraw
p: print statement
q: quit
w
Enter account number
3334
Enter amount to withdraw
75
Account : 3334Your current balance is 225.56
Options
d: deposit
w: withdraw
p: print statement
q: quit
p
Enter account number
3334
Bank Account 3334 Statement Date :Tue Mar 26 17:58:23 EDT 2013 Current balance :
225.56
Deposit : Tue Mar 26 17:57:29 EDT 2013 amount : 200.0 End balance : 300.56
```

```
Withdrawal : Tue Mar 26 17:58:14 EDT 2013 amount : -75.0 End balance : 225.56

Options
d: deposit
w: withdraw
p: print statement
q: quit
w
Enter account number
2223
Enter amount to withdraw
50
Warning : Your balance is under $0. A fee of $25 will be applied
Account : 2223Your current balance is -75.0
Options
d: deposit
w: withdraw
p: print statement
q: quit
p
Enter account number
2223
Bank Account 2223 Statement Date :Tue Mar 26 17:58:55 EDT 2013 Current balance : -
75.0
Error : Tue Mar 26 17:58:41 EDT 2013 ERR2556B - insufficient funds  amount : 50.0
account 2223
Withdrawal : Tue Mar 26 17:58:41 EDT 2013 amount : -50.0 End balance : -75.0

Options
d: deposit
w: withdraw
p: print statement
q: quit
p
Enter account number
7777
The account was not found
Options
d: deposit
w: withdraw
p: print statement
q: quit
q
Bye!
```

Your classes should include Javadoc style commenting, and be properly formatted. Make sure that your classes enforce proper visibility. This is due on April 24, NO LATE SUBMISSIONS ACCEPTED! I need time to grade them! This grade will substitute for your project1 grade if it is higher. If you did well on project1, you do not need to do this project.