CMS 270 Object Oriented Design and Development Fall 2020

Inheritance & Polymorphism

submit via Canvas. **Please DO NOT email them to me.**

# You may submit as many times as you wish, before the deadline. Only your last submission will be graded, so ensure that this is your best version

**Objectives:** The purpose of this question, is for students to demonstrate knowledge of

* Inheritance and polymorphism
* How inheritance works in Java
* how to build classes and sub-classes in Java
* How to read data from text files
* How to write data to text files
* How to use classes to create and manipulate objects

**Problem:**

You are required to simulate a batch processing system for a small, local Credit Union. The Credit Union manages 2 types of accounts: Savings, Checking accounts. All accounts have a number, an owner and a balance. Savings accounts have a minimum balance and a maintenance fee. The minimum balance and maintenance fee for Savings account is the same for all Savings Accounts. You may assume that it is 100.00 and 25.00 respectively. If the balance in the savings account ever goes below the minimum, the maintenance fee is charged once per month, until the situation is corrected. Checking accounts have a last check issued, and last check used and a maximum number of checks allowed for the month. Transactions are executed on an account. A transaction has a type (which may be withdraw, transfer, deposit or close)

Identify the classes that you need and their relationships. Code each of your classes including all of your getters and setters and any other helper methods that you think will be useful to solve the problem.

You may assume that you have 2 files available – ***accounts.txt*** that contains information about the current status of all accounts; and ***batch.txt*** that contains a set of transactions to be performed on the accounts. The transactions in batch.txt are expected to reflect one month’s worth of transactions

Create a class called ***BatchProcessor***. BatchProcessor will be your driver class.

When the driver is run, your program should read the original account-status data from ***accounts.txt***, and store all of the accounts in a ***single*** ArrayList. It should then read ***batch.txt*** and process the transactions one at a time applying them to the relevant accounts. After processing all of the transactions in the batch file, your program should write the updated account status data to the text file ***accounts.txt***. This should overwrite the original file.

**The Accounts.txt file is formatted with each Account’s data on a new line**. Each line:

* begins with the account number
* then a letter indicating the account type (*C* for checking, *S* for savings)
* then the name of the owner- Given as firstname lastname
* then the account balance
* For checking accounts the line continues with the number of the last check issued, followed by the last check used and then the maximum checks allowed for the month – all of which are integers.

***Batch.txt is formatted such that each line contains information for a single transaction***. The line

begins with a single letter (W, D, T or C) indicating the type of transaction. Then:

* For lines beginning with W, the line of data will continue with the account number followed by the amount to be withdrawn, then the name of the person requesting the withdrawal.
* For lines beginning with D, the line of data will continue with the account number followed by the amount to be deposited
* For lines beginning with T, the line of data will continue with the account number for the source of the Transfer, then the account number for the destination of the transfer and the name of person requesting the transfer.
* For lines beginning with C, the line of data will continue with the account number for the account to be closed and the name of the person requesting the closure.

All accounts should be able to handle each type of transaction. They should all override the methods in Account. The transaction methods should not be overloaded.

**Note that there may be multiple transactions on the same account and they do not have to be in sequence.**

**Some program pointers:**

You may assume that all data in the input files is formatted correctly.

* + Withdrawals will only be permitted for transactions requested by a legitimate owner of the account.
  + All of the rules for withdrawals also apply for the transfers.
  + For an account closing to be allowed:
    - the balance must be positive
    - the legitimate owner must request the closure o When an account is closed, the account record is deleted from the company’s records o *The driver class must define and use the following methods:*
    - main
    - ***Account processDeposit(Account a,double Amt)***
    - ***Account processWithdrawal(Account a, double Amt, String owner)***
    - ***Account processTransfer(Account a, Account b, double Amt, String owner)***
    - ***Account processClose(Account a, String owner)***
    - When a transaction is not approved, an appropriate message should be printed to screen.
    - You may include other helper functions in this class or any other class if you wish
    - For each class there should be at least one constructor that initializes all of the data members.

## Note: All of the input files have a single space delimiter between each of the elements of data

**Sample Run**

Sample accounts.txt

123 C Mary Cross 50000.00 200 154 24

211 S Andrew Green 2500.00

|  |
| --- |
| *Output on screen after program has been run*  Account 211 could not be closed  *Accounts.txt after program has been run*  123 C Mary Cross 50100.00 200 154 24  211 S Andrew Green 1600.00 |

145 S Vaughn Hope 75000.00

Sample batch.txt

D 145 50.25

W 211 800.00 Andrew Green

D 145 72.00

C 145 Vaughn Hope

C 211 Mary Cross

T 211 123 100.00 Andrew Green