**LA7 Sunset Banking System**

|  |  |
| --- | --- |
| **Lab Time** | **Due Date** |
| **Wednesday** | **04/09/2014** |
| **Thursday** | **04/10/2013** |

**Background**

Sunset Banking System is a kind of system used for Banks. During the daytime, the bankers can submit transactions, for instance withdraw transactions or deposit transactions, to the system. But the system will just keep the transactions without processing them which means they are not going to be reflected in the account balance immediately. The system processes all the transactions at the end of each day. That is why it is called Sunset Banking System. It only processes transactions at the time of Sunset.

**Problem Specification**

You are to write a program in Java to implement this Sunset Banking System. You are given the data of all checking accounts and all the transactions. You job is to process all the transactions and have them reflected in the balance of the accounts. For example if there is a deposit transaction to the account 100010 with the amount of 100 dollars, then the balance of account 100010 should be increased by 100. It works the other way around for withdraw transactions. You are given three input files.

1. account.txt

This file contains the data of all the banking accounts. Each line contains the information for one account. Each line contains 5 tokens in the order: accounts number, holder’s first name, holder’s last name holder’s phone number and the account balance.

Example

100001 Ala Al-Fuqaha 276-3868 10032.32

100002 Steve Carr 276-3101 34234234.45

100003 Ajay Gupta 276-3104 3435.45

100004 Donna Kaminski 276-3104 44545.4

1. withdraw.txt

This files contains the data of all the withdraw transactions. Each line contains the information for one withdraw transaction. Each line contains two numbers account number and amount of this transaction.

Example

100001 100

100002 200

100003 300

100004 400

100005 500

1. deposit.txt

This files contains the data of all the deposit transactions. Each line represents one deposit transaction. Each line contains two numbers the account number and the amount of this transaction.

Example

100006 100

100007 200

100008 300

100009 400

100010 500

**After processing all the transactions, you program should print out all the accounts info to the terminal. Than we should be able to see changes in the balance due to those transactions.**

**Design Requirement**

You are required to have the following classes

/\*

\* This class is the super class of DepositTransaction and WithdrawTransaction

\*/

**public** **class** Transaction {

**int** accountNumber;

Date date;

**double** balanceChange;

**void** process(Account[] accounts){}

}

**TransactionList class is a generic class used to store transactions. It can operate on both deposit transactions and withdraw transaction. You need to implement the following generic class yourself. Write your own class generic TransactionList<T> You are not allowed to use Java class. Implement the following methods yourself.**

**public** **class** TransactionList<T **extends** Transaction> {

// **add more attributes or methods if needed.**

**void** process(Account[] accounts){}

**void** add(T transaction){}// Add a element to the list

T get(**int** index){} // return the element with specific index

**int** size(){} // return the size of the list.

}

**You have to implement the following class for the file input and out**.

**public** **class** FileInput {

**public** Account[] loadAccounts(String fileName) **throws** FileNotFoundException{

// **TODO** Auto-generated method stub

**return** **null**;

}

**public** TransactionList<Withdraw> loadWithdrawTransactions(String fileName) {

// **TODO** Auto-generated method stub

**return** **null**;

}

**public** TransactionList<Withdraw> loadDepositTransaction(String fileName) {

// **TODO** Auto-generated method stub

**return** **null**;

}

}

**You need to implement Accounts class with methods as used in Main().**

**public** **class** Account{/\*…………\*/}

**Test**

Test your program with the following class.

**public** **class** Test {

**public** **static** **void** main(String[] args) **throws** FileNotFoundException {

FileInput fileReader = **new** FileInput();

Account[] accounts = fileReader.loadAccounts("account.txt");

//for(int i=0;i<accounts.size();i++) System.out.println(accounts.get(i));

TransactionList<Withdraw> withdraws = fileReader.loadWithdrawTransactions("withdraw.txt");

withdraws.process(accounts);

TransactionList<Deposit> deposits = fileReader.loadDepositTransaction("deposit.txt");

deposits.process(accounts);

**for**(**int** i=0;i<accounts.length;i++){

System.*out*.print(accounts[i].getAccountNumber()+" ");

System.*out*.print(accounts[i].getHolderFirstName()+" ");

System.*out*.print(accounts[i].getHolderLastName()+" ");

System.*out*.print(accounts[i].getHolderPhoneNumber()+" ");

System.*out*.print(accounts[i].getBalance());

System.*out*.println();

}

}

}

**Submission**

* Generate a .zip file that contains all your files, including:
  + program source files
  + any input, and
  + any output files
* Upload the zipped file to eLearning.