CISC 327 Course Project

Assignment #4

Back end: rapid prototype

Company: Canada Trust

Yudong Zhou (20083467)

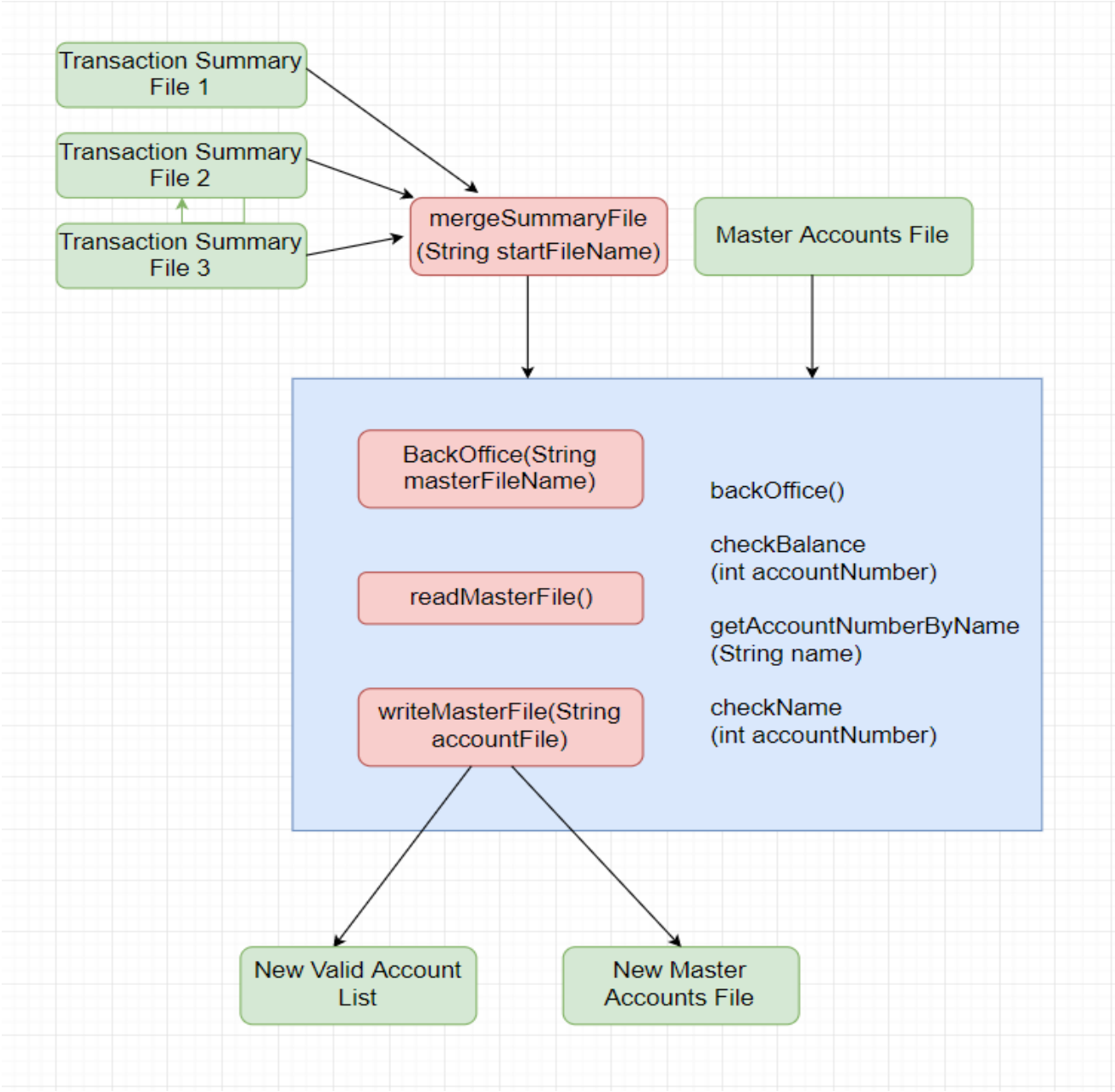
Yitong Liu (20028039)

Runze Yi (20073329)

Tong Bu (20079649)

Overall structure

The overall structure of our solution is shown in the picture below. We use the blue boxes to represent the main class of our project while the green boxes represent input and output files.
The Back Office class read the master account file and the transaction summary files generated from different front ends as the two input files. It writes the new Master Accounts file and new Valid Account List file summary file as an output file. The main functions in the class are colored into pink, and other small functions are also listed inside the blue box.



Classes and methods

The table below shows all classes and methods in our source code followed by a brief description for each of them.

Class Name: BackOffice	
Function Name	**Description**
BackOffice()	This is the non-parameter constructor for BackOffice class
BackOffice(String masterFileName)	This is the constructor that creates BackOffice object by taking the master account file as the parameter and starts to read the master file. Each object will have 3 attributes, one is the masterFileName as String, one is Hashmap that links each available account number to each account balance, one is Hashmap that links each account number to each account name.
mergeSummaryFile(String startFileName)	This method takes the transaction summary file name prefix as the parameter, starts to read the all summary transaction files in the current directory(for example "SummaryTransaction" as the prefix, and this method will read all prefix of SummaryTransaction file in the folder, such as SummaryTransaction1, SummaryTransaction2, SummaryTransaction3 and etc…)
writeMasterFile()	This method writes out the new master accounts file, and also write the new updated valid account list file.
readMasterFile()	This method reads the master accounts file and will update two new attributes: one is Hashmap that links each available account number to each account balance, one is Hashmap that links each account number to each account name.
checkBalance (int accountNumber)	This method will take a accountNumber as the parameter, and return that account balance by this account number (It is used to check some accounts balance by the front end, for example, when we try to delete an account from the frontend, we have to check this account balance by backoffice if it is a balance of zero, otherwise, we can't delete it)
checkName (int accountNumber)	This method takes an accountNumber as the parameter, this method will return an account name by certain account number. It makes sure the user deletes the correct account with the exact name due to security. (For example, when we try to delete an account from the frontend, that user wants to delete his account of 1234567 Jack, but he accidentally typed another valid account name, this will occur delete the wrong account, or information not matches issue. We have to make sure that user's input of delete account number and account name is matched)

Summary

Our program will start running by the following command at the console:

javac Start.java

java Start AccountList SummaryFile MasterFile

“AccountList SummaryFile MasterFile” will be 3 arguments

AccountList is the accountList file name(input file)

SummaryFile is the summary transaction file name that you want the system to write for(output file)

Master file is the master account file name (input file, and it is also will be output file since the back office will update it)