FrontEnd – Main method of the program, lines of text are read looking for the “login” command, after a successful login command the class “session” will be created. All lines of the text that do correspond to a proper log on will be ignored.

FrontEnd.checkForLogin() – keeps checking the text stream for a login command, ignoring everything else. Subsequently looks for “atm” or “agent” if “login” is found, after a proper login a session class is created.

Class Session – A class that is used to create a session after a successful login, the session has a Boolean so that the program can know if it is agent or atm access. Once created lines of text will be read, and if they are a valid command the corresponding method within the class will be invoked. The class contains a class object for ValidAccounts as well as a class of Transactions for the transactionlist.

Session.withdrawDeposit(bool deposit) - invoked whenever the withdraw/deposit command are used. Combined into one method as the checks for withdraw and deposit are the exact same, the only difference is if the money is being removed/added which is known by the argument “bool deposit”, if the value is false then the amount will be withdrawn. Checks to see if the account and amount are valid (by calling checkValidAmount and checkValidAccount).

Session.logout() - when called the transaction summary file is written, and then the session class is destroyed (going back to the main until a successful login occurs once again)

Session.transfer() – similar to withdrawDeposit() once called asks the user for from account number, to account number, and amount to transfer and then checks to see if the accounts and amount are valid (calling checkValidAmount on the value and checkValid account on the accounts).

Session.create() – command is denied if not in agent mode, in agent mode asks for the account number and name and does the necessary checks on the constraints.

Session.delete() - command is denied if not in agent mode, in agent mode asks for the account number and name and does the necessary checks on the constraints. If successful the account is added to the invalid account list for use in the current front end session.

Session.writeTransaction() – called during Session.logout(), all transactions recorded in the session are written to the transaction summary file (done by classing Class.writeTransactionSummaryFile)

Session.readAccounts() – creates a ValidAccounts class and then reads in the valid account file, done on creation of the session class.

Session.checkValidAmount(int amount) – checks to see if the amount is between 0-$1000 for atm mode, and 0-$999,999.99 if in agent mode

Session.command(string input) – chooses which command function to call when given the input from the textstream

Class ValidAccounts – a class that contains all of the valid accounts read in from the valid account file in an array

ValidAccounts.checkValidAccont(int accountNumber) – checks if the given account number exists

Class Transactions – a class created on creation of a session to hold all of the transactions done, contains an array “transactionsList” which contains all of the transactions in chronological order. (transactionsList[0] is the first transactions, etc)

Transactions.addTransaction(string TransactionName) – Adds a transaction to the existing transaction list

Transactions.writeTransactionSummaryFile() – writes everything in the transactionList to the transaction summary file