**Description of the Intention of Each Method and Class**

**Akira Aida - 100526064**

**Kathryn McKay - 100524201**

**Alexander Wheadon - 100514985**

|  |  |
| --- | --- |
| **Method or Class** | **Description** |
| Class Account | Stores all account information. |
| Class Transaction | Stores all information about a transaction. |
| Class TransactionType | References transaction codes used in transaction files |
| Class FileParser | Parse the files and puts them into data structures to easily manipulate them. Concatenates multiple transaction files into one and puts it into a list. |
| Map<String, ArrayList<Account>> parseMaster(String masterFile) | Read in master accounts file, parses it and return the accounts in the file as a map with the account holder's name as a string and an arraylist of accounts (with the account information for each) as the value. |
| void concatTrans(List<String> transFiles) | Create one transaction file by combining existing transaction files by reading them in and putting their contents inside a new file concat.txt. |
| List<Transaction> parseTrans | Read in a list of transactions from the file that concatTrans creates, parses the transactions and returns them as a list of transactions. |
| Class FileUpdater | Takes the data structure (name->their accounts) that has had the transactions applied to it and writes the new master accounts' file. |
| void fileWriter(Map<String, ArrayList<Account>> accounts) | Loops through the data structure (name->their accounts) and writes all of the accounts to a "newMaster.txt" file using the createAccount helper function |
| String createAccount(String name, int num, boolean stat, double bal, int trans, boolean plan) | A helper function for the fileWriter which formats a string (the account) to be written to a file |
| Class TransactionCalculator | Handles and applies transactions to the specified accounts. |
| void setAccountTable(Map<String, ArrayList<Account>> account\_table) | Gives known accounts to the calculator for reference. |
| Map<String, ArrayList<Account>> getAccountTable() | Retrieves account map with all of accounts in it. |
| void applyTransactions(Vector<Transaction> transactions) | Selected the appropriate transaction handler for the selected transaction. |
| Account getAccount(Transaction transaction) | Searches for account in the bank system by looking for the holder's name and then account number |
| double getTransactionFee(Account account) | Based on the type of account the transaction is being performed on give specified transaction fees. (admin/student/non-student) |
| boolean accountNumberExists(int number) | Check if account number exist in the data structure of accounts. |
| void handleLogin(Vector<Transaction> transactions) | Processes the 'login' transaction. |
| void handleLogout(Vector<Transaction> transactions) | Processes the 'logout' transaction. |
| void handleWithdrawal(Vector<Transaction> transactions) | Perform Withdrawal transaction to the specified account. |
| void handleTransfer(Vector<Transaction> transactions) | Perform Transfer transaction to the specified accounts (sender and receiver). |
| void handlePaybill(Vector<Transaction> transactions) | Perform Pay bill transaction to the specified account. |
| void handleDeposit(Vector<Transaction> transactions) | Perform Deposit transaction to the specified account. |
| void handleChangePlan(Vector<Transaction> transactions) | Perform Change Plan transaction to the specified account. |
| void handleDelete(Vector<Transaction> transactions) | Remove the specified account from the account list. |
| void handleCreate(Vector<Transaction> transactions) | Add new account the account list if the account number is not already in use. |
| void handleDisable(Vector<Transaction> transactions) | If the account is not already disabled, disable the specified account. |
| void handleEnable(Vector<Transaction> transactions) | If the account is not already enabled, enable the specified account. |
| Class Backend | The class with the main method in it which will take in file inputs as arguments and execute all driver code to create a new current accounts file and a new master accounts file. |
| static void main(String[] args) | The program will take a master accounts file and one or many transaction files and then create a new master accounts file with the transactions applied to it from the old master accounts file |