**I. Show the methods invoked for building the collection of clients**

// added a setData member function to Client to make reading in data

// easier

firm::addClients(infile){

while still lines in file {

Declare Client object pointer  //Client ptr = new Client;

Set ptr data                    //ptr->setData(infile);

Insert Client into Binary Tree  //binaryTree.insert(client)

}

}

client:: setData(infile) {

Read accountID and ten account values from clientFile/infile

}

1. **USE CASE 1**

**firm::addTransactions(infile){**

**Declare Transaction object pointer;    // Trns ptr = new Trns;**

**while file is not empty {**

**Set ptr data        // ptr->setData(infile);**

**Enqueue transaction ptr**

**}**

**}**

// set data for transactions

// could alternatively use a switch instead of if-else branches

**transactions::setData(infile) {**

**Declare Transaction object;**

**if(the first char in the line from infile is ‘D’) {**

**Use infile to read in Transaction object data memebers;**

**Enqueu into queue**

**}**

Else if(the first char in the line from infile is ‘W’) {

Use infile to read in Withdraw object data memebers;

Enqueu into queue

}

Else if(the first char in the line from infile is ‘M’) {

Use infile to read in Transaction object data memebers;

// amountWithdraw, amountDeposit, command, clientID

Enqueu into queue

}

Else if (the first char in the line from infile is ‘D’) {

Use infile to read in Transaction object data memebers;

Enqueu into queue

}

Else {

Error message

}

}

}

**firm::processTransactions(){**

**while transaction remaining in transactions queue {**

**Dequeue transaction from queue**

**if(transaction command is ‘D’) {**

**Retrieve client from binary search tree using clientID**

**Perform deposit on Client**

**// find account from account array in client object**

**}**

Else if(transaction command is ‘W’) {

Retrieve client from binary search tree using clientID

Perform deposit Client

}

Else if(transaction command is ‘M’) {

Find first client using getFrom()

Find second client using getTo

Find amount of money to move   // getAmount();

}

Else if(transaction command is ‘H’) {

Retrieve client from binary search tree using clientID

Show Client history

}

}

}

1. **USE CASE 2**

// added a setData function into Transaction class to make reading from

// file easier

// was not sure if this is the best way to implement

**firm::addTransactions(infile){**

**Declare Transaction object pointer;    // Trns ptr = new Trns;**

**while file is not empty {**

**Set ptr data        // ptr->setData(infile);**

**Enqueue transaction ptr**

**}**

**}**

// set data for transactions

// could alternatively use a switch instead of if-else branches

**transactions::setData(infile) {**

**Declare Transaction object;**

if(the first char in the line from infile is ‘D’) {

Use infile to read in Transaction object data memebers;

Enqueu into queue

}

Else if(the first char in the line from infile is ‘W’) {

Use infile to read in Withdraw object data memebers;

Enqueu into queue

}

**Else if(the first char in the line from infile is ‘M’) {**

**Use infile to read in Transaction object data memebers;**

**// amountWithdraw, amountDeposit, command, clientID**

**Enqueu into queue**

**}**

Else if (the first char in the line from infile is ‘D’) {

Use infile to read in Transaction object data memebers;

Enqueu into queue

}

Else {

Error message

}

}

}

**firm::processTransactions(){**

**while transaction remaining in transactions queue {**

**Dequeue transaction from queue**

if(transaction command is ‘D’) {

Retrieve client from binary search tree using clientID

Perform deposit Client depending on account number

// find account from account array in client object

}

Else if(transaction command is ‘W’) {

Retrieve client from binary search tree using clientID

Perform deposit Client

}

**Else if(transaction command is ‘M’) {**

**Find first client using getFrom() // use firstAccountID**

**Find second client using getTo()  // use secondAccountID**

**Find amount of money to move using getAmount();**

**// Client uses withdraw()**

**Withdraw money depending on account number of 1st client**

**// Client uses deposit()**

**Deposit money depending on account number of 2nd client**

**}**

Else if(transaction command is ‘H’) {

Retrieve client from binary search tree using clientID

Show Client history

}

}

}