CPS707 Assignment 5

Team Name: Team 10

Armen Palvetzian
Louise Marquez
Prachi Kapadia

Code Listing

dailyTransactionWriter():

```
public static String dailyTransactionWriter(String type, String userName, String accType,
double credit) throws IOException {
                    File file = new
File("C:/Users/Armen/Desktop/CPS707/test-files/daily-transactions.txt");
                    FileWriter fr = new FileWriter(file, true);
                    BufferedWriter br = new BufferedWriter(fr);
                    br.newLine();
                    br.write(type + "_" + userName + "_" + accType + "_" + credit +"\n");
                    br.close();
                    fr.close();
                    String test = type + "_" + userName + "_" + accType + "_" +
                    credit; return test;
          }
```

updateUserFile():

```
public static String updateUserFile(ArrayList<User> userArray, String user, String
transactionFile, String userFile) {

                    try {
                              PrintWriter writer = new
PrintWriter("C:/Users/Armen/Desktop/CPS707/test-files/newUserFile.txt");
                              writer.close();

                              File file = new
File("C:/Users/Armen/Desktop/CPS707/test-files/newUserFile.txt");
                              FileWriter fr = new FileWriter(file, true);
                              BufferedWriter br = new BufferedWriter(fr);

                              /*
                              for(User u : userArray) {
```

```
                                        br.write(u.getUserName() + "_" + u.getAccountType() + "_" +
u.getBalance());
                                        br.newLine();
                              }*/
                              br.close();
                              fr.close();

                    }catch (IOException e) {}

                    return "" + user + "_" + transactionFile + "_" + userFile;

          }
```

createUser():

```
public static String createUser(String userName, String newuserName, double credit, String
accType) throws IOException{

String result = "";

          userArray.add(new User ("pkapadia", "AA", 200.0));
          //String newuserName;
          /*System.out.println("Enter a new username:  or 0 to cancel");
          Scanner scan = new Scanner(System.in);
          newuserName = scan.nextLine();
          */

          //Checks to see if the username is more than 15 characters
          if(newuserName.length() > 15) {
                    System.out.println("Too many characters! Please try again");
                    result =  "Too long";
                    return result;
                    //createUser();
          }

          //Checks to see if the balance is more than 999,999
          if(credit > 999999){
                    System.out.println("The balance is too high please try again");
                    //createUser();
                    result = "The balance is too high please try again";
                    return result;
          }

          //Checks to see if the user already exists
          for(User u : userArray) {
                              if(u.getUserName().equals(newuserName)) {
                              System.out.println("User already exists!");
                              return "User already exists!";
                              //createUser();
                    //break;
                    }
            }

        //Cancel option
        if(newuserName.equals("0")){
                adminInterface();
        }
        else {
                //String accType;
                //double credit;
                String creditstr; //the string format of the credit in the account
                Scanner scan2 = new Scanner(System.in);

                if (accType != null && !accType.isEmpty()) {
                        System.out.println("Type of user (AA, SS, BS): ");
                        result = "AA";
                        return result;
                }

        System.out.println("Set the balance amount: ");
        creditstr = scan2.nextLine();
        //credit = Double.parseDouble(creditstr);

        userArray.add(new User(newuserName, accType, credit));
        dailyTransactionWriter("01", userName, "AA", balance);
        updateUserFile(userArray, userName, "newTicket.txt", "newUserFile.txt");

        System.out.println("New user added!");
        for(User u : userArray) {
                System.out.println(u.getUserName() +  " " + u.getAccountType() + " " +
u.getBalance() );
        }

        result = "01" + "_" + userName + "_" + "AA" + balance;
        adminInterface();

        }
        return result;

        }
```

<u>refund():</u>

```
public static String refund(String buyerName, String sellerName, double balanced)
throws IOException{
            String result = "";
            String buyer;//The buyer who wants a refund
            String amount;//The in string format
            double amountD;//The amount in double format

            double amountS = 0;//The sellers balance;

            System.out.println("Buyer's username (Refund to): ");
            Scanner scan = new Scanner(System.in); //buyer =
            scan.nextLine();
            buyer = buyerName;

        //Checks to see if the user exists
      for(User u : userArray) {
            u.userName = buyer;
            if(!userArray.contains(u)) {
                  System.out.println("Buyer does not exist!");
                  //refund();
                  result = "Buyer does not exist!";
                  return result;
                  refund();
            }
      }

            System.out.println("Seller's username (Refund from): ");
            Scanner scan2 = new Scanner(System.in); //seller =
            scan2.nextLine();
            seller = sellerName;

            System.out.println("Amount being refunded: ");
            Scanner scan3 = new Scanner(System.in);
            //amount = scan3.nextLine();
            //amountD = Double.parseDouble(amount);
            //amountS -= amountD;

            //balance += amountD;
            if(balanced > 0) {
                  result = "Accepatable Amount!";

            }
            amountD = balanced;
        //Corrects the balance amounts for both accounts

            System.out.println(buyer + " received a " + amountD + " refund from " + seller);
            dailyTransactionWriter("05", buyer, seller, amountD);
            updateUserFile(userArray, userName, "ticket.txt", "login.txt"); //return "05_" +
            buyer + "_" + seller + "100";
            return result;
```

Test Cases

WhiteBox Method Testing	**Testing Transaction**	**Test Case Name**	**Intention**
Statement Coverage	Account Creation	testCreate1	Ensures that once all the new user information is given that it is accurately written out in the transaction file and it updates the user file
Statement Coverage	Account Creation	testCreate2	Ensures that the user exist statement is reached and nothing gets printed in the transaction file
Statement Coverage	Account Creation	testCreate3	Tests that the user name is not greater than 15 characters.
Statement Coverage	Account Creation	testCreate4	Tests that the user is able to set the account type of the created user.
Statement Coverage	Account Creation	testCreate5	Tests that when the

			max credit limit is reached that the transaction isn’t written to the backend
Decision Coverage Testing	Ticket Cancellation	testTicket1	Tests that when the user does not exist the refund transaction stops. When the user name is valid it continues with the refund transaction
Decision Coverage Testing	Ticket Cancellation	testTicket2	Tests that when the balance is greater than 0 the refund transaction is complete. Otherwise the transaction ends.

Decision Coverage:

Test Case Name	**Decision Coverage**	**Tests True**	**Tests False**
testTicket1	Line: 664 If (!userArray.contains(u) (FrontEnd-refund.java)	The username does not exist hence the transaction stops with the refund	Test transaction should include and continue with the refund. Input: apalvetz_AA_100
testTicket2	Line: 701 If (balanced>0) (FrontEnd-refund.java)	Test transaction should not include a 05 in the header and should not continue with the refund. Input: 0	Test transaction file should include the balance and 05. Input: 05_user1_user2_100.00
testTicket3	Line: 112 in BackEnd.java return "" + user + "_" + transactionFile + "_" + userFile;	The user file is updated with the user information	The user file is not updated with the user information
testTicket4	Line: 706 in FrontEnd.java if(balanced > 0) {	When the amount is greater than 0, the amount is acceptable.	When the amount is less than 0, the amount is not acceptable.

Statement Coverage:

Test Case Name	**Statement Coverage**	**Tests True**	**Tests False**

testCreate1	Line: 31 - 32 in BackEnd.java
	String test = "" + type + "_" + userName + "_" + accType + "_" + "" + credit;
	return test;
		New user daily transaction log is written out.	Doesn't write out the daily transaction file

testCreate2	Line: 321 – 329 in FrontEnd.java
	for(User u : userArray)
	if(u.getUserName().equals(newuserName))
	System.out.println("User already exists!");
	return "User already exists!";
		New user is created and added to the user ArrayList	User already exists in the ArrayList

testCreate3	Line: 305 – 311 in FrontEnd.java
	if (newuserName.length() > 15)
	System.out.println("Too many characters! Please try again");
	result = "Too long";
	return result;
		When username is greater than 15 characters	When username is less than 15 characters

testCreate4	Line: 341 – 345 in FrontEnd.java
	if (accType != null && !accType.isEmpty())
	System.out.println("Type of user (AA, SS, BS): ");
	Scanner scan2 = new Scanner(System.in);
	result = "AA";
	return result;
		When account type of the user is set.	When we can’t set user account

testCreate5	Line: 313 – 319 in FrontEnd.java if (credit > 999999) System.out.println("The balance is too high please try again"); result = "The balance is too high please try again"; return result;	When balance is too high. Greater than 999,999	When balance is less than 999,999

Test Report

No errors were reported. All the tests were successful.