Assignment 5 - Back Office Unit Testing

ZoopDeeBoop

Meryl Gamboa - 10173192 - 2 hours - Documented the test cases
Zane Little - 10179568 - 2 hours - Analyzed the test cases
Stefan Urosevic - 10146785 - 2 hours - Analyzed the test cases

Testing createacct using statement testing

Stmt	Master Accounts Input	Test Input	Master Accounts Output	What Is It Testing?	Error
211	1111111 50000 Zane 2222222 000 Stef EOF	NEW 1234567 000 0000000 foobar EOS	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that statement 1 runs. This is checked by making sure the account number is correct. 1234567 is the correct number	N/A
212	1111111 50000 Zane 2222222 000 Stef EOF	NEW 1234567 000 0000000 foobar EOS	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that statement 2 runs. This is checked by making sure the balance is correct. 000 is the correct balance	N/A
213	1111111 50000 Zane 2222222 000 Stef EOF	NEW 1234567 000 0000000 foobar EOS	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that statement 3 runs. This is checked by making sure the account name is correct. “foobar” is the correct name	N/A
214	1111111 50000 Zane 2222222 000 Stef EOF	NEW 1234567 000 0000000 foobar EOS	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	Check that the account is inserted in order (statement 4). This is checked by making sure that foobar is the second account in the	N/A

Assignment 5 - Back Office Unit Testing

				master accounts file (Ascending account number)	

```
207  // upNEW takes accounts vector and a transaction string
208  // It returns accounts vector updated for withdraw
209  vector<string> upNEW(vector<string> accounts, string transaction, int acctPos)
210  {
211      string newAcct = transaction.substr(4, 7);
212      newAcct += " 000 ";
213      newAcct += transaction.substr(24, transaction.length() - 24);
214      accounts = insertInOrder(accounts, newAcct);
215      return accounts;
216  }
```

Testing withdraw using decision coverage

Stmt	**Master Accounts Input**	**Test Input**	**Master Accounts Output**	**What Is It Testing?**	**Error**
169: True	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	XFR 1234567 50 1111111 *** EOS	1111111 49950 Zane 1234567 50 foobar 2222222 000 Stef EOF	Test that the program enters the transfer specific block when the transaction code is XFR.	N/A
169: False	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that the program does not enter the transfer specific block when the transaction code is WDR.	N/A

Assignment 5 - Back Office Unit Testing

172: True	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	XFR 3333333 50 1111111 *** EOS	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	Tests that the program catches if an account does not exist.	N/A
172: False	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	XFR 1234567 50 1111111 *** EOS	1111111 49950 Zane 1234567 50 foobar 2222222 000 Stef EOF	Test if the if statement can be false if the account exists.	N/A
180: True	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef EOF	Tests the while loop that gets the transaction amount.	N/A
180: False	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that it exits the while loop and doesn’t stall.	N/A
188: True	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef EOF	Tests the while loop that gets the transaction amount.	N/A
188: False	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef	Test that it exits the while loop and doesn’t stall.	N/A

Assignment 5 - Back Office Unit Testing

	EOF		EOF		
199: True	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	XFR 1234567 50 1111111 *** EOS	1111111 49950 Zane 1234567 50 foobar 2222222 000 Stef EOF	Tests the if statement that only is true on transfer transactions.	N/A
199: False	1111111 50000 Zane 1234567 000 foobar 2222222 000 Stef EOF	WDR 1111111 50 0000000 ***	1111111 49950 Zane 1234567 000 foobar 2222222 000 Stef EOF	Test that it doesn’t go into the if statement during withdraw transactions.	N/A

Assignment 5 - Back Office Unit Testing

```
157  // upWDR takes accounts vector and a transaction string
158  // It returns accounts vector updated for withdraw
159  vector<string> upWDR_XFR(vector<string> accounts, string transaction, int acctPos)
160  {
161      int x = 12;
162      string transAmt = "";
163      string oldAmt = "";
164      int newAmt = 0;
165      int acctPos2;
166
167
168      // Find account position for
169      if (transaction[0] == 'X')
170      {
171          acctPos2 = findPos(accounts, transaction.substr(4, 7));
172          if (acctPos2 == -1)
173          {
174              writeError("ERROR WITH XFR : ACCOUNT NUMBER " + transaction.substr(4,7) + " NOT FOUND IN ACCOUNTS FILE.ACCOUNTS MANIFEST NOT UPDATED.", false);
175              return accounts;
176          }
177      }
178
179      //get transaction amount
180      while (transaction[x] != ' ')
181      {
182          transAmt += transaction[x];
183          x++;
184      }
185
186      //get previous amount
187      x = 8;
188      while (accounts[acctPos][x] != ' ')
189      {
190          oldAmt += accounts[acctPos][x];
191          x++;
192      }
193
194      string oldAcct = accounts[acctPos];
195      //calculate new ammount
196      newAmt = stoi(oldAmt) - stoi(transAmt);
197      accounts[acctPos] = oldAcct.substr(0, 7) + " " + to_string(newAmt) + " " + findAcctName(oldAcct, 8);
198      // deal with transfer further by updating the 'to' account
199      if (transaction[0] == 'X')
200      {
201          accounts = upDEP(accounts, transaction, acctPos2);
202      }
203      return accounts;
204  }
```