**Akira Aida - 100526064**

**Kathryn McKay - 100524201**

**Alexander Wheadon - 100514985**

|  |  |  |
| --- | --- | --- |
| **Test Failure** | **Test Case** | **Test Error Message** |
| No END\_OF\_FILE in the new master accounts file | public void testOneWriter(): Tests that fileWriter correctly outputs a master accounts file and currents accounts file given one account. Both files need to end with END\_OF\_FILE | testWriter(tests.TestUpdater): expected:<[11111 Bob A 00005.23 0034 S, 00000 END\_OF\_FILE A 00000.00 0000 S]> but was:<[11111 Bob A 00005.23 0034 S]> |
| No END\_OF\_FILE in the new current accounts file | public void testOneWriter(): Tests that fileWriter correctly outputs a master accounts file and currents accounts file given one account. Both files need to end with END\_OF\_FILE | testWriter(tests.TestUpdater): expected:<[11111 Bob A 00005.23 S , 00000 END\_OF\_FILE A 00000.00 S ]> but was:<[11111 Bob A 00005.23 S]> |
| Test stub 'getAccount' was looking up accounts not ordinally but by account number; in this case looking for the 12345th index in an array. | public void checkHandleChangePlan():  Tests that the handleChangePlan function can run in it's entirety and with graceful handling of error conditions. | checkHandleChangePlan(tests.TestTransactionCalculator): java.lang.IndexOutOfBoundsException: Index: 12345, Size: 5 |
| handleDeposit does not print error message for negative balance case | public void checkHandleDeposit():  Verifies whether error conditions of deposit can be met. | checkHandleDeposit(tests.TestTransactionCalculator): Expected error to occur? [] expected:<true> but was:<false> |
| Besides the formatting error in the expecting output, no error is printed at all by handleWithdrawal for the negative case. | public void checkHandleWithdrawal():  Verifies whether progression through withdrawal function is as expected. | checkHandleWithdrawal(tests.TestTransactionCalculator): expected:<[ERROR: Account %d balance is negative]> but was:<[]> |
| handleDisable does not correctly handle nonexistent/deleted accounts. | Public void checkHandleDisable():  verifies whether checking for special cases proceeds as expected. | checkHandleDisable(tests.TestTransactionCalculator): java.lang.NullPointerException |
| handleCreate does not correctly handle nonexistent/deleted accounts. | Public void checkHandleCreate():  verifies whether checking for special cases proceeds as expected. | checkHandleCreate(tests.TestTransactionCalculator): java.lang.NullPointerException |
| handleDelete does not correctly handle nonexistent/deleted accounts. | Public void checkHandleDelete():  verifies whether checking for special cases proceeds as expected. | checkHandleDelete(tests.TestTransactionCalculator): java.lang.NullPointerException |
| handleEnable does not correctly handle nonexistent/deleted accounts. | public void checkHandleEnable():  verifies whether checking for special cases proceeds as expected. | checkHandleEnable(tests.TestTransactionCalculator): java.lang.NullPointerException |
| Propagation of how handleCreate does not correctly handle nonexistent/deleted accounts. | Public void checkAccountNumberExists():  determines whether functions have coverage of different account number existence cases. | checkAccountNumberExists(tests.TestTransactionCalculator): java.lang.NullPointerException |
| handleLogin does not throw error for bad misc flag. | public void checkHandleLogin():  Verifies that different procedures for handling invalid logins go through. | checkHandleLogin(tests.TestTransactionCalculator): Unexpected console contents ''' wanted error? expected:<false> but was:<true> |
| handlePaybill is not showing error for negative balance. | public void checkHandlePaybill():  Verifies that different procedures for handling invalid transactions go through. | checkHandlePaybill(tests.TestTransactionCalculator): expected:<[ERROR: Account %d balance is negative]> but was:<[]> |
| There is no checking for valid transaction sequences. | public void checkApplyTransactions():  Verifies that all code is used in transaction processing. | checkApplyTransactions(tests.TestTransactionCalculator): Unexpected error message expected:<[ERROR: Transaction sequence had no logout after login]> but was:<[]> |
| There is no error output for invalid transfer sequences. | public void checkHandleTransfer():  Verifies that different procedures for handling invalid transactions go through. | checkHandleTransfer(tests.TestTransactionCalculator): expected:<[ERROR: Amount transferred and amount sent do not match: $5.00 vs $105.00]> but was:<[]> |
| handleWithdrawal has no handling for accounts that are not found. | public void checkGetAccount():  Verifies that existing and nonexisting accounts are handled. | checkGetAccount(tests.TestTransactionCalculator): java.lang.NullPointerException |
| Also a victim of handleWithdrawal's lack of accounts not found handling | public void checkGetTransactionfee:  Verifies that the different fees can be returned. | checkGetTransactionFee(tests.TestTransactionCalculator): java.lang.NullPointerException |