**Akira Aida - 100526064**

**Kathryn McKay - 100524201**

**Alexander Wheadon - 100514985**

|  |  |  |
| --- | --- | --- |
| **Name of Test Case** | **Scenario of Test Case** | **Expected Results** |
| public void testActStu() | Tests the parseMaster method with an input that's an active as well as a student account.  \*Statement coverage  \*Decision coverage for true  \*Loop coverage for one iteration | The account is correctly parsed with the account's setting being the expected results. With A for active = true. And S for student = true. |
| public void testDisNon() | Tests the parseMaster method with an input that's a disabled as well as a non-student.  \*Statement coverage  \*Decision coverage for false  \*Loop coverage for one iteration | The account is correctly parsed with the account's setting being the expected results. With D for disabled = false. And N for non-student = false. |
| public void testMultiple() | Tests the parseMaster method with a user that has multiple accounts.  \*Statement coverage  \*Loop coverage for multiple iterations | The accounts are correctly parsed with the account's settings being the expected results. The ArrayList data structure correctly holds all of the accounts for this user with the map key being the name. |
| public void testEnd() | Tests the parseMaster method with an input that's the END\_OF\_FILE.  \*Statement coverage  \*Loop coverage for no iterations | The outputted file only has the special END\_OF\_FILE account and no other accounts in it. |
| public void testOneConcat() | Tests one file being concatenated allows for no issues.  \*Statement coverage  \*Loop coverage for one iteration | The outputted concatenated file has the contents of one transaction file |
| public void testThreeConcat() | Tests three files are being concatenated.  \*Statement coverage  \*Loop coverage for three iterations | The outputted concatenated file has the contents of the three transaction files |
| public void testTrans() | Tests that a transaction was correctly parsed.  \*Statement coverage | The transaction is correctly parsed into its respective attributes. |
| 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  \*Statement coverage  \*Loop coverage for one iteration | The current accounts file has the one user and all of its’ accounts details with the END\_OF\_FILE at the end of it. The master accounts file has the one user and all of its’ accounts details with the END\_OF\_FILE. The difference between the two files is the master has the number of transactions. |
| public void testEmptyWriter() | Test that an empty accounts data structure produces an END\_OF\_FILE file  \*Statement coverage  \*Loop coverage for no iterations | Both files just have the END\_OF\_FILE account at the end with the master having spaces allotted for the number of transactions. |
| public void testCurrActS() | Check current account is created correct that's active and a student  \*Statement coverage  \*Decision coverage for true | The string that's created follows the format of the current account's file and has A for active and S for student. |
| public void testCurrDisN() | Check current account is created correct that's disabled and a non-student  \*Statement coverage  \*Decision coverage for false | The string that's created follows the format of the current account's file and has D for disabled and N for non-student. |
| public void testMastActS() | Check master account is created correct that's active and a student  \*Statement coverage  \*Decision coverage for true | The string that's created follows the format of the master account's file and has A for active and S for student. |
| public void testMastDisN() | Check master account is created correct that's disabled and a non-student  \*Statement coverage  \*Decision coverage for false | The string that's created follows the format of the master account's file and has D for disabled and N for non-student. |
| public void testProper() | Test the main method with proper arguments which executes the entire program and shows that the main driver code is all executed.  \*Statement coverage | The three files output all have the correct contents. |
| public void checkApplyTransactions() | Check that all correct transactions are applied and incorrect ones create errors  \*Statement coverage | Absence and presence of error messages and by examining the accounts file to be sure the transactions were applied |
| public void checkGetAccount() | Check that specified account exist and if it can be modified  \*Loop coverage  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkGetTransactionFee() | Check transactions are applied to the specified accounts based on type  \*Loop coverage  \*Statement coverage  \*Decision coverage | Check transaction files for difference in transaction based on the fees for each account |
| public void checkAccountNumberExists() | Check that the presented account number doesn’t exist if it does throw error  \*Loop coverage  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleLogin() | Check for correct input formatting  \*Loop coverage  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleLogout() | Check that logout is processed  \*Statement coverage | Check transactions list that originally contained logout transaction is now empty |
| public void checkHandleWithdrawal() | Check that withdrawal is successful if the balance in larger than the withdrawal otherwise it will throw an error  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleTransfer() | Check that transfer is successful if the balance in larger than the transfer amount and if the send money is equal to the received otherwise it will throw an error  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandlePaybill() | Check that pay bill is successful if the balance in larger than the pay bill otherwise it will throw an error  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleDeposit() | Deposit 5 cents as admin and user to check transaction  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| checkHandleChangePlan() | Check correct parameters from the change accounts function  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleDelete() | Checks for existence of account to be deleted  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleCreate() | Check if the account to be created account’s number already exists.  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleDisable() | Check if account is not disabled otherwise throw error  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |
| public void checkHandleEnable() | Check if account is disable otherwise throw error  \*Statement coverage  \*Decision coverage | Absence and presence of error messages |