Delirium Apple

Phase 1: Test Case Design

February 11th, 2016
Version 1.1
9 Pages

Nicholas Gregorio - 100514374
Janahan Mathamohan - 100523201
Matthew McCormick - 100448975

Test Organization

We have organized our tests into two main folders. One folder contains the input, and the other folder contains the expected terminal output. Inside the two main folders are folders that will test a either a specific transaction, general input into the terminal, or the transaction files created. These folders are named to correspond with the specific command they test.

Each file in the INPUT folder contains commands the terminal will take, each on a new line. Some of these commands are valid, while others are invalid. Each file in the OUTPUT folder contains either the expected terminal output, each on a new line, or the expected transaction file to be created in the case of the TRANSACTIONS and BANKACCOUNTS folder.

A complete table of all tests is below:

Test Descriptions

Folder	**File**	**Description**
DEPOSIT	DepositAdminSuccessful.txt	Deposit a valid amount of money to a valid standard account as an admin.
DEPOSIT	DepositDiffAccounts.txt	Ensures a user with multiple accounts can deposit to each of those in a single session
DEPOSIT	DepositDisabled.txt	Attempt to deposit to an account that is disabled. Checks that an error message is displayed to the user.
DEPOSIT	DepositInvalidAccountNumber.txt	Attempts to deposit to an account with a valid account name but incorrect account number. Also attempts to deposit to an account with an invalid account name. Checks that an error message is displayed to the user.
DEPOSIT	DepositInvalidAccountNumberAdmin.txt	Attempts to deposit to an account with an invalid account number as an admin. Checks that an error message is displayed to the user.
DEPOSIT	DepositNonMatchingAccountNumber.txt	Attempts to deposit to an account with a valid account name and the account number from a different account. Checks that an error message is displayed to the user.
DEPOSIT	DepositNonMatchingAccountNumberAdmin.txt	Attempts to deposit to an account with a valid account name and the account number from a different account as an admin. Checks that an error

		message is displayed to the user.
DEPOSIT	DepositUpperLimit.txt	Attempts to deposit an amount that is greater than the daily limit as a standard user and as an admin. Checks that an error message is displayed to the user.
DEPOSIT	InvalidSessionFunds.txt	Ensures that deposited funds are not readily available to the user.
DEPOSIT	SuccessfulDeposit.txt	Deposit a valid amount of money to a valid standard account as a standard user.
DEPOSIT	ValidSessionFunds.txt	Ensures balance is still correctly shown after a deposit followed by a withdrawal
DISABLE	DisableAccountAlreadyDisabled.txt	Disables an account that was previously disabled. Checks that an error message is displayed to the user.
DISABLE	DisableAccountAsStandard.txt	Attempts to disable an account from a standard account. Checks that an error message is displayed to the user.
DISABLE	DisableAccountInvalidName.txt	Attempts to disable an account with an invalid name. Checks that an error message is displayed to the user.
DISABLE	DisableAccountInvalidNumber.txt	Attempts to disable an account with a valid name and invalid account number. Checks that an error message is displayed to the user.
DISABLE	DisableAccountSuccessful.txt	Disable an account with a valid name and account number. Checks that the transaction is written to the transaction file.
ENABLE	EnableAccountAlreadyEnabled.txt	Enables an account that was previously enabled. Checks that an error message is displayed to the user.
ENABLE	EnableAccountAsStandard.txt	Attempts to enable an account from a standard account. Checks that an error message is displayed to the user.
ENABLE	EnableAccountInvalidName.txt	Attempts to enable an account with an invalid name. Checks that an error message is displayed to the user.

ENABLE	EnableAccountInvalidNumber.txt	Attempts to enable an account with a valid name and invalid account number. Checks that an error message is displayed to the user.
ENABLE	EnableAccountSuccessful.txt	Enable an account with a valid name and account number. Checks that the transaction is written to the transaction file.
INITALCOMMANDS	InvalidPreLoginCommandType.txt	Attempt to run invalid commands before logging in. Checks that an error message is displayed to the user.
INITIALCOMMAND	ValidCommandLoggedOut.txt	Attempt to run valid commands before logging in. Checks that an error message is displayed to the user.
LOGIN	DoubleLogin.txt	Attempts to login when already logged in. Checks that an error message is displayed to the user.
LOGIN	InvalidLoginAccountType.txt	Attempts to login with an invalid account type. Checks that an error message is displayed to the user.
LOGIN	InvalidLoginAdmin.txt	Attempts to login as an admin with invalid name. Checks that an error message is displayed to the user.
LOGIN	InvalidLoginStandard.txt	Attempts to login as a standard user with invalid name. Checks that an error message is displayed to the user.
LOGIN	InvalidToValidLoginAccountType.txt	Attempts to login as a standard user with an invalid account type then logs in with valid information. Checks that the user is still able to log in.
LOGIN	InvalidToValidLoginAdmin.txt	Attempts to login as an admin with an invalid name then logs in with valid information. Checks that the user is still able to log in.
LOGIN	InvalidToValidLoginStandard.txt	Attempts to login as a standard user with an invalid name then logs in with valid information. Checks that the user is still able to log in.
LOGIN	NewLoginSuccessful.txt	Logs into an account, logs out, then logs into the same account again. Checks that login and logout work correctly.
LOGIN	ValidLoginAdmin.txt	Logs in as an admin. Tests that login as admin works correctly.

LOGIN	ValidLoginStandard.txt	Logs in as a standard user. Tests that login as a standard user works correctly.
PAYBILL	PayBillAdminInvalidAccount.txt	Attempt to pay a bill to an invalid account as an admin. Checks that an error message is displayed to the user.
PAYBILL	PayBillAdminInvalidCompany.txt	Attempt to pay a bill to an invalid company as an admin. Checks that an error message is displayed to the user.
PAYBILL	PayBillAdminInvalidFunds.txt	Attempt to pay a bill to a valid company with an invalid amount as an admin. Checks that an error message is displayed to the user.
PAYBILL	PayBillAdminValid.txt	Pay bills for multiple accounts to multiple companies as an admin. Checks that the bills were paid.
PAYBILL	PayBillDisabled.txt	Attempt to pay a bill from a disabled account. Checks that an error message is displayed to the user.
PAYBILL	PayBillStandardInvalidAccount.txt	Attempt to pay a bill to an invalid account as a standard user. Checks that an error message is displayed to the user.
PAYBILL	PayBillStandardInvalidCompany.txt	Attempt to pay a bill to an invalid company as a standard user. Checks that an error message is displayed to the user
PAYBILL	PayBillStandardInvalidFunds.txt	Attempt to pay a bill to a valid company with an invalid amount as a standard user. Checks that an error message is displayed to the user.
PAYBILL	PayBillStandardLimit.txt	Attempts to pay bills with a total amount more than the daily limit. Attempts to pay bills with a single amount more than the daily limit. Checks that an error message is displayed to the user.
PAYBILL	PayBillStandardValid.txt	Pay bills to multiple companies as a standard user. Checks that the bills were paid.
TRANSFER	TransferAdminInvalidBalance.txt	Attempt to transfer an invalid amount as an admin. Checks that an error message is displayed to the user.
TRANSFER	TransferAdminNonMatching.txt	Attempt to transfer to an account with valid name and invalid account number. Checks that an error

		message is displayed to the user.
TRANSFER	TransferAdminValid.txt	Transfer a valid amount to a valid account as an admin. Checks that the amount was transferred to the correct account.
TRANSFER	TransferDisabled.txt	Attempt to transfer from a disabled account. Checks that an error message is displayed to the user.
TRANSFER	TransferStandardInvalidAccount.txt	Attempt to transfer to to an invalid account as a standard user. Checks that an error message is displayed to the user.
TRANSFER	TransferStandardLimit.txt	Attempt to transfer amounts that are greater than the dailylimit. Checks that an error message is displayed to the user.
TRANSFER	TransferStandardValid.txt	Transfer a valid amount to a valid account as a standard user. Checks that the amount was transferred to the correct account.
TRANSFER	TransferUpperLimit.txt	Attempts to transfer the maximum amount allowed in an account as an admin and as a standard user. Checks that an error message is displayed to the user.
WITHDRAW	InvalidAdminWithdrawalAccount.txt	Attempt to withdraw from an account with an invalid name and an invalid account number as an admin. Checks that an error message is displayed to the user.
WITHDRAW	InvalidAdminWithdrawalAmounts.txt	Attempts to withdraw from a valid account with an invalid amount as an admin. Checks that an error message is displayed to the user.
WITHDRAW	InvalidStandardWithdrawalAmount.txt	Attempts to withdraw from a valid account with an invalid amount as a standard user. Checks that an error message is displayed to the user.
WITHDRAW	InvalidStandardWithdrawlAccount.txt	Attempt to withdraw from an account with an invalid name and an invalid account number as a standard user. Checks that an error message is displayed to the user.
WITHDRAW	StandardWithdrawalLimit.txt	Attempt to withdraw amounts greater than the daily limit as a standard user. Checks that an error message is displayed to the user.
WITHDRAW	StandardWithdrawalLimi	Attempt to withdraw amounts greater than the daily

	tSession.txt	limit in multiple sessions as a standard user. Checks that an error message is displayed to the user.
WITHDRAW	StandardWithdrawlDiffAccounts.txt	Attempt to withdraw amounts from multiple accounts greater than the daily limit. Checks that an error message is displayed to the user.
WITHDRAW	ValidAdminWithdrawal.txt	Withdraw a valid amount from a valid account as an admin. Checks that the amount was withdrawn from the correct account.
WITHDRAW	ValidStandardWithdrawal.txt	Withdraw a valid amount from a valid account as a standard user. Checks that the amount was withdrawn from the correct account.
WITHDRAW	WithdrawDisabled.txt	Attempt to withdraw from an account that is disabled. Checks that an error message is displayed to the user.
TRANSACTION	LoginAdminTransactionCreated.txt	Attempts to create a transaction file and tests that the admin login processes Transaction codes are in it.
TRANSACTION	LoginStandardTransactionCreated.txt	Attempts to create a transaction file and tests that the standard login processes Transaction codes are in it.
TRANSACTION	WithdrawalTransactionCreated.txt	Attempts to create a transaction file and tests that the withdrawal processes Transaction codes are in it.
TRANSACTION	PaybillTransactionCreated.txt	Attempts to create a transaction file and tests that the paybill processes Transaction codes are in it.
TRANSACTION	TransferTransactionCreated.txt	Attempts to create a transaction file and tests that the transfer processes Transaction codes are in it.
TRANSACTION	DepositTransactionCreated.txt	Attempts to create a transaction file and tests that the deposit processes Transaction codes are in it.
TRANSACTION	CreateTransactionCreated.txt	Attempts to create a transaction file and tests that the create processes Transaction codes are in it.
TRANSACTION	DeleteTransactionCreated.txt	Attempts to create a transaction file and tests that the delete processes Transaction codes are in it.
TRANSACTION	ChangeplanTransactionCreated.txt	Attempts to create a transaction file and tests that the changeplan processes Transaction codes are in it.
TRANSACTION	DisableAccountCreated.txt	Attempts to create a transaction file and tests that the disable process transaction codes are in it.

TRANSACTION	EnableAccountCreated.txt	Attempts to create a transaction file and tests that the enable process transaction codes are in it.
TRANSACTION	MultipleAccountHolders.txt	Attempts to create a transaction file with multiple account holders and transactions being put to one file
TRANSACTION	MultipleAccountNumbers.txt	Attempts to create a transaction file with multiple account numbers and transactions being put to one file
BANKACCOUNT	CreateBankAccount.txt	Attempts to make changes to the bank account file with the command create.
BANKACCOUNT	ChangeplanBankAccount.txt	Attempts to make changes to the bank account file with the command changeplan.
BANKACCOUNT	DisableBankAccount.txt	Attempts to make changes to the bank account file with the command disable.
BANKACCOUNT	EnableBankAccount.txt	Attempts to make changes to the bank account file with the command enable.
BANKACCOUNT	WithdrawalAccount.txt	Attempts to make changes to the bank account file with the command withdrawal.
BANKACCOUNT	DepositBankAccount.txt	Attempts to make changes to the bank account file with the command deposit.
BANKACCOUNT	DeleteBankAccount.txt	Attempts to make changes to the bank account file with the command delete.

Testing Procedure

We will use a Bash script(subject to change to possible two) to read a list of tests, run the tests, and compare the output and transaction file to the expected files. The script will write any discrepancies found into a single file with information about what tests caused the issues.

In addition to this script, the front end will be populated with accounts that we will use for testing purposes. These will have various account numbers and balances, and will use different plans and statuses:

- **Troy Bartan**.
    - Account number: 10000. Balance: $500.00.
    - Account number: 10001. Balance: $1200.00
    - Student Plan on both
- **Phil Cooper**.
    - Account number: 20000. Balance: $100

    - Student Plan
- **Timothy Franklin**.
    - Account number: 10002. Balance $99999.99
    - Non-student Plan
- **George Frick**.
    - Account number: 15000 Balance $250
    - Non-student Plan
    - DISABLED
- **Jose Ankin**: to be deleted and recreated
- **Sam Fischer**
    - Account number: 11000. Balance: $2000. Student plan
    - Account number: 11001. Balance: $3000. Non-student plan

Testing Order

Initially, we will run the INITIALCOMMANDS tests in order to ensure the front end will not accept any incorrect inputs that could cause errors in the subsequent test cases. Following this, the test cases regarding LOGIN (and logout) will be tested, as these are the fundamental commands that will be run in every test afterwards. The WITHDRAWAL and DEPOSIT tests will then be run, starting by alternating between a withdrawal test and a deposit test. This is to maintain account balances.

- ValidStandardWithdraw, then SuccessfulDeposit (maintain balance of $500.00)
- StandardWithdrawalLimit (uses deposit commands, should maintain balances of $1200.00 and $500.00)
- StandardWithdrawalLimitSession, then ValidSessionFunds (in Deposit, should maintain balance of $500.00)
- StandardWithdrawalDiffAccounts, then DepositDiffAccounts (maintain balance)
- ValidAdminWithdrawal followed by DepositAdminSuccessful (maintain balance)

This sequence will then be followed by the remaining withdrawal and deposit tests. These are invalid tests and should not have an effect on the balance of the test accounts.

Following this, the PAYBILL, TRANSFER, & CHANGEPLAN transactions will be tested. ENABLE will be tested after, and must be tested before DISABLE (running subsequently after enable) as the test cases will enable an account that is initially disabled, and disable will disable this account again. Following these test cases, the last transaction tests CREATE & DELETE will be run.

After all transaction commands are run, the actual resultant transaction files will be tested. These will use their own set of inputs in order to test these in isolation. Testing transactions separately after testing the specific commands will allow us to more accurately test file creation.

Requirements Ambiguities and Assumptions:

The following ambiguities were found during this phase of production, in the form of the ambiguity, the action taken, and what the system will do in relation to this ambiguity:

- Unclear if disabled accounts can still be on the receiving end of a transfer. Clarification requested, and disabled accounts will not be allowed to receive transfers

- Unclear how new accounts are created. Clarification requested. New accounts will be made by entering an unused name during the CREATE transaction.
- Unclear if users should be shown their balance after both successful and unsuccessful transactions. Implementation detail. Balances will be shown.
- Unclear on minimum amount of transfers. Clarification requested. All transfers will be greater than $0.
- Unclear on how to cancel a current transaction. Clarification requested. Invalid input during a transaction will take the user to the last, successfully completed transaction.