**Test Plan**

**Akira Aida – 100526064**

**Kathryn McKay – 100524201**

**Alexander Wheadon – 100514985**

**Test Input and Output Organization**

- Input files will be containing the inputs that each test case needs to run. Each input will be on a newline representing a command being entered.

- Two output files will contain the outputs that each test case is expecting (these are what the test should output). One file will contain the console output, with each output on a newline. The other will contain the transactions that take place for that specific test case, each will also be on a newline.

- Two files will be created for each test case. One to capture the console outputs and one to capture the transactions, both will be “diff”'d to the corresponding output files to create the pass/fail file. If the output of the “diff” for both is nothing then the test case passes, otherwise it fails.

- A Pass/Fail file will be created with the date/time when the entire test completes and states which test cases passed and failed for that build. This will contain the same information that the script will output to the terminal with the pass/fails. Thus allowing the pass/fails to be easily assessed in the future and for easy comparisons to previous builds.

- e.g. 10Feb0237.pass representing the 10th of February at 2:37am.

- All of the testing files will be in a directory for testing (which will be on the same level as the project directory).

- The testing directory will contain; the script, the temporary files that are created for each test case (that will be deleted after each test case), the input directory and the output directory.

- The input files will be stored in subdirectories for their specific function that they are testing.

- The output files will be stored in subdirectories for their specific function that they will be testing.

- The pass/fail files will be stored in a directory so that they can be referenced if needed in the future.

**Test Run Plan**

- There will be one bash script that will run every test case.

- The script will run each test case by running the program with the accounts file.

- Each test case will use one of the input files that are located in the input directory (Directory is divided into subfolders by function) and feed them into the program.

- The script will pipe the console outputs into a temporary file for comparison after completion.

- The script will save the transactions file.

- After completion, the console outputs file and the transaction file will be “diff”'d to the .out and .trans file that correspond with the .in file.

- If it returns a difference then the test case fails, otherwise it passes.

- If there's an error then the test case fails immediately.

- The output of the test case after completion will be sent to the console as well as appended to a file.

- The file will store all the pass/fails for each test case for that build and will be created at the start of a build.

- The temporary files will then be deleted so that they do not affect any future test cases.

The directory structure will be as follows...

Directory structure:

├── project/

│ └── src/

│ └── resources/

└── tests/

├── inputs/

│ ├── chng/

│ ├── crte/

│ ├── delt/

│ ├── depo/

│ ├── dsbl/

│ ├── enab/

│ ├── logn/

│ ├── logt/

│ ├── payb/

│ ├── tran/

│ └── with/

├── outputs/

│ ├── chng/

│ ├── crte/

│ ├── delt/

│ ├── depo/

│ ├── dsbl/

│ ├── enab/

│ ├── logn/

│ ├── logt/

│ ├── payb/

│ ├── tran/

│ └── with/

└── passes\_fails/

With the contents of each directory being...

Contents of each directory:

- wrapper/

- project/

- .exe file

- src/

- source files

- resources/

- transactions.txt

- accounts.txt

- tests/

- script file

- two temporary file during a test case that captures outputs

- inputs/

- subdirectories for functionality points/

- input files .in

- outputs/

- subdirectories for functionality points/

- expected output files .out

- expected transaction files .trans

- passes\_fails/

- pass/fail files .pass/.fail