**TEST PLAN**

**TEST INPUT AND OUTPUT ORGANIZATION**

We have created a directory named test\_case\_files that contains all the tests’ input and output files; these files are then be grouped and divided into the different commands to be tested (login, withdrawal, transfer, paybill, deposit, create, delete, disable, changeplan, and logout) – except for the current\_bank\_accounts file, which is also part of this directory.

Two additional commands were created: help and quit. help will aid the users remember which commands are available - as well as their details -, while quit will enable the user to exit the system.

As mentioned above, each of this commands will be given a separate directory. Inside each of these directories, there will be four more: input, expected\_output, real\_output and transaction. The input, expected\_output and transaction directories contain pre-prepared testing files, whereas the real\_output directory will contain the actual results of the test when the program is ran.

With regards to naming conventions, we decided to use .in, .out and .tra for input, output and transaction files, respectively. Each was named using the command they are meant to be testing, followed by a number – which corresponds to the step being tested on the test cases – and then a letter, which represents one of the variations for that specific test (e.g. login05b.in).

**TEST RUN PLAN**

The tests created will be ran on our Banking System using a shell script, who’s output will be stored in the real\_output directory, as mentioned above, and compared with the expected\_output files using the diff command.

The output files and the bank account transaction files that resulted from the input files being ran in the program will be compared with the expected output files and expected bank account transaction files.

If the outputs do not match, or if the bank account transaction files do not match, then the test will show false. When false is shown, there is a known problem within the program, and the program is not producing the correct results. If output matches, and the bank account transaction files match, then the test will show true. When true is shown, there program is producing the correct results.

All the tests in the sub-group will be run with a script, which will show all the tests within the sub-group with the result of true or false. This can be used to fix the sub-group problems and the scope of the problem can be narrowed down. When all the tests in the sub-group come out with all true results, the sub-group is producing all the correct results. The script will be run with every sub-group. When all tests from every sub-group come out with all true results. The whole program is producing the correct results.

The order in which the sub-groups are run matters as some tests from other sub-groups depend on the working conditions of the other sub-groups. The first two sub-groups that will be tested will be quit and help. These two are tested first as the program needs a way to exit gracefully, and if a user is confused, they could always refer to the help menu. The next two sub-groups that will be tested will be login and logout. These two should be tested next as login and logout will be used in every test case. Next will be followed by create and delete. Create and delete as the following transaction test cases need to test if an account has been just created. Following create and delete will be disable and enable. These two are tested as changeplan, deposit, withdrawl, transfer, and paybill all have test cases that depend on whether an account is disabled or active. The next transaction to be tested would be changeplan as deposit, withdrawl, transfer, and paybill all have fees that depend on the plan of the user.

Next test would be for deposit. This is tested next as withdrawal, transfer, and paybill all need deposit to test if the account can be used for those transactions. Finally, withdrawal, transfer and paybill will be tested. These are tested last as these 3 sub-groups depend on the sub-groups tested before them.

luisarojas [test\_case\_files] $ tree .

.

├── changeplan

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── create

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── current\_bank\_accounts\_file.txt

├── delete

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── deposit

│   ├── input

│   ├── real\_output

│   └── transaction

├── disable

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── enable

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── help

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── login

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── logout

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── paybill

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── quit

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

├── transfer

│   ├── expected\_output

│   ├── input

│   ├── real\_output

│   └── transaction

└── withdrawal

├── expected\_output

├── input

├── real\_output

└── transaction