Assignment 6 – Integration and Delivery

Group 19: Meara Donovan, Bruce Nishimura, Sam McPhail

Daily Script:

```
1.  import glob
2.  import os
3.  import sys
4.  import tempfile
5.
6.  import backend
7.  import frontend
8.
9.  # simulates 3 separate sessions for 1 day
10. def main(number_of_sessions=3):
11.     for i in range(number_of_sessions):  # run the front end instance 3 times with
    appropriate inputs
12.         # prepare program parameters
13.         sys.argv = [
14.             'frontend.py',
15.             'valid_accounts.txt',
16.             'new_transactions/transaction_' + str(i) + '.txt']
17.         frontend.main()
18.     temp_fd, temp_file = tempfile.mkstemp()
19.     merged_transaction_file = temp_file
20.     transaction_files = glob.glob("new_transactions/*.txt")  # list of transaction summary
    files
21.     with open(merged_transaction_file, 'w') as wf:  # create merged txn summary file
22.         for file in transaction_files:
23.             with open(file, 'r') as tf:
24.                 for line in tf:
25.                     if line != 'EOS 0000000 000 0000000 ***':
26.                         wf.write(line)
27.         wf.write('EOS 0000000 000 0000000 ***')
28.     for transaction in transaction_files:
29.         os.remove(transaction)
30.     # run backend with updated files
31.     sys.argv = [
32.         'backend.py',
```

```
        'master_accounts.txt',
        merged_transaction_file]
    backend.main()
    os.close(temp_fd)
    os.remove(temp_file)

if __name__ == "__main__":
    main()
```

Weekly Script:

```
import io
import sys
import daily

# runs the daily script 5 times, providing the correct input
def main():
    # at the start of the week, start with an empty accounts list
    with open('valid_accounts.txt', 'w') as wf:
        wf.write('0000000')
    open('master_accounts.txt', 'w').close()  # wipe master accounts file
    days = [
        # day 1
        ['login', 'agent', 'createacct', '1001001', 'Coolest account', 'logout',
         'login', 'agent', 'createacct', '2222222', 'Acct 2', 'logout',
         'login', 'agent', 'createacct', '3333333', 'Acct 3', 'logout'],
        # day 2
        ['login', 'agent', 'createacct', '1111111', 'Acct 1', 'logout',
         'login', 'machine', 'deposit', '1001001', '33300', 'logout',
         'login', 'machine', 'transfer', '1001001', '3333333', '33300', 'logout'],
        # day 3
        ['login', 'machine', 'deposit', '2222222', '20000000', '200000', 'logout',
         'login', 'machine', 'deposit', '2222222', '200000', 'logout',
         'login', 'machine', 'deposit', '2222222', '100000', 'logout'],
        # day 4
        ['login', 'machine', 'deposit', '1111111', '1700', 'logout',
         'login', 'machine', 'withdraw', '3333333', '100100', '1000', 'logout',
         'login', 'agent', 'deleteacct', '1001001', 'Coolest account', 'logout',
         'login', 'agent', 'createacct', '123456', '1234567', 'hi', 'give me your money', 'logout'],
        # day 5
```

```
        ['login', 'machine', 'transfer', '1111111', '1234567', '1700', 'logout',
         'login', 'machine', 'transfer', '2222222', '1234567', '500000', 'logout',
         'login', 'machine', 'transfer', '3333333', '1234567', '32300', 'logout']]

    # run daily script for each day detailed above
    for i in range(5):
        sys.stdin = io.StringIO(
            '\n'.join(days[i]))
        if i != 3:
            daily.main()
        else:
            daily.main(4)

if __name__ == "__main__":
    main()
```

Day 2: Daily Run – Session Inputs and File Contents:

Session 1:

```
login
agent
createacct
1111111
Acct 1
logout
```

Session 2:

```
login
machine
deposit
1001001
33300
logout
```

Session 3:

```
login
machine
transfer
1001001
3333333
33300
```

```
logout
```

Merged Transaction Summary file:

```
NEW 1111111 000 0000000 Acct 1
DEP 1001001 33300 0000000 ***
XFR 3333333 33300 1001001 ***
EOS 0000000 000 0000000 ***
```

Master Accounts File Contents:

Day 1:

```
3333333 0 Acct 3
2222222 0 Acct 2
1001001 0 Coolest account
```

Day 2:

```
3333333 33300 Acct 3
2222222 0 Acct 2
1111111 0 Acct 1
1001001 0 Coolest account
```

Day 3:

```
3333333 33300 Acct 3
2222222 500000 Acct 2
1111111 0 Acct 1
1001001 0 Coolest account
```

Day 4:

```
3333333 32300 Acct 3
2222222 500000 Acct 2
1234567 0 give me all your money
1111111 1700 Acct 1
```

Day 5:

```
3333333 0 Acct 3
2222222 0 Acct 2
1234567 534000 give me all your money
1111111 0 Acct 1
```

Integration defect report:

All in all, the integration between our front and backend went very smoothly. We found only a few not previously caught issues. Only one of the defects found was related to the integration;

we had to modify logout so that it ended the session altogether, in order to simulate daily runs. The defects are detailed in the table below:

Problem	Description	Solution
End program	Logging out should end the program altogether. Our implementation has a quit command that ends the session, but this didn't prevent someone from logging out and logging back in again, which would result in the transaction summary file being overwritten.	To avoid this problem, we changed logout so that the session is ended upon logout
Spaces in account name	We noticed that the backend parses the master accounts list file and merged transaction summary file based on spaces, which is an issue if the account names contain spaces.	We modified the backend so that spaces in account names are accounted for and accurately preserved.
Delete account	When finding the account to delete, the backend was trying to access the account number at the index of the respective account balance. We were therefore unable to delete the account, as we didn't have the correct number.	The backend now correctly checks for the account number when looking for the account to delete.

Note regarding daily limits:
We don't have a way to check transactions from other sessions in the front end, so if the daily deposit amount, for example, exceeds the $5000 limit, but all deposits are made validly during different sessions, we are unable to see that the limit was exceeded.. The daily limit checking was not listed as a backend requirement, so we interpreted it as being a limit for a day at a given ATM (one session), and implemented this as such.

Weekly Script inputs:
Day 1:

```
['login', 'agent', 'createacct', '1001001', 'Coolest account',
'logout']
['login', 'agent', 'createacct', '2222222', 'Acct 2', 'logout']
['login', 'agent', 'createacct', '3333333', 'Acct 3', 'logout']
```

Day 2:

```
['login', 'agent', 'createacct', '1111111', 'Acct 1', 'logout']
['login', 'machine', 'deposit', '1001001', '33300', 'logout']
['login', 'machine', 'transfer', '1001001', '3333333', '33300',
'logout']
```

Day 3:

```
['login', 'machine', 'deposit', '2222222', '20000000',
'200000','logout']
['login', 'machine', 'deposit', '2222222', '200000','logout']
['login', 'machine', 'deposit', '2222222', '100000', 'logout']
```

Day 4:

```
['login', 'machine', 'deposit', '1111111', '1700'', 'logout']
['login', 'machine', 'withdraw', '3333333', '100100', '1000',
'logout']
['login', 'agent', 'deleteacct', '1001001', 'Coolest account',
'logout']
['login', 'agent', 'createacct', '123456', '1234567', 'hi', 'give
me your money', 'logout']
```

Day 5:

```
['login', 'machine', 'transfer', '1111111', '1234567', '1700',
'logout']
['login', 'machine', 'transfer', '2222222', '1234567', '500000',
'logout']
['login', 'machine', 'transfer', '3333333', '1234567', '32300',
'logout']
```