Assignment 6 – Armaan Merchant, Pradnya Sadhu Putra, Tony Chen, Tyler Elliott

Source listing of Daily Script:

```
import os
import subprocess
import sys
import time
import random

merged_tsf = open("mergedTransactionSummaryFile.txt", "a+")

def merge_tsf_files(final):
    """Combine the transaction summary file generated for the session with all others from
the same day

    :param final: True if this is the last session of a day, False otherwise
    :return: None
    """
    tsf_file = open("transactionSummaryFile.txt", "r")
    contents = tsf_file.read()
    merged_tsf.write(contents)
    if final:
        merged_tsf.write("EOS")


def get_random_command():
    """Using a list of commands, return a random item from the list

    :return: A command for the front end
    """
    commands = [
    "deposit\n1234567\n900000\n",
    "deposit\n7654321\n550000\n",
    "deposit\n1234566\n1000000\n",
    "deposit\n1234567\n10000\n",
    "deposit\n7654321\n9999\n",
    "createacct\n1234566\narmin\n",
    "createacct\n1234567\ngod\n",
    "createacct\n7654321\nblueface\n",
    "transfer\n1234566\n1234567\n33300\n",
    "transfer\n1234567\n7654321\n66600\n",
    "transfer\n1234566\n7654321\n90000\n",
    "transfer\n7654321\n1234567\n4000\n",
    "transfer\n7654321\n1234566\n8800\n",
    "withdraw\n1234567\n100000\n",
    "withdraw\n7654321\n54321\n",
    "transfer\n1234567\n7654321\n696969\n",
    "deposit\n7654321\n99999\n",
    "createacct\n6969696\nnice\n",
    "deleteacct\n6969699\nnice\n",
    "transfer\n1234567\n6969696\n111111\n",
    "deleteacct\n69\ngeazy\n",
    "createacct\n6969696\ngeazy\n",
    "deposit\n1234567\n50000\n",
    "deposit\n6969696\n999999\n",
```

```
    "createacct\n6955569\npleaseGiveUsS\n",
    "withdraw\n6969696\n500\n",
    "deposit\n7654321\n1111111\n",
    "withdraw\n7654321\n50000\n",
    "withdraw\n1234567\n50000\n"
    ]

    return random.choice(commands)


def get_random_sequence(length):
    """Generate and format a string of random commands

    :param length: The number of commands that should be generated
    :return: A formatted list with (length) commands """

    string = "login\nagent\n"
    for y in range(length):
        string += get_random_command()
    string += "logout"
    return string


def first_day():
    """The first day should only create accounts

    :return: None
    """

    # Clear the merged tsf file from last session
    merged_tsf.truncate(0)
    print("--- Day 1 ---")
    # Compile the java files so they can be run from command line
    subprocess.run("javac src/main/java/*.java")

    print("Ran session 1 automatically")
    subprocess.run("java -cp src/main/java FrontEnd
validAccountList.txt transactionSummaryFile.txt",

input="login\nagent\ncreateacct\n1234566\narmin\ncreateacct\n1234567\ngod\nlogout", text=True)
    merge_tsf_files(False)

    print("Ran session 2 automatically")
    subprocess.run("java -cp src/main/java FrontEnd
validAccountList.txt transactionSummaryFile.txt",
                   input="login\nagent\ncreateacct\n7654321\nblueface\nlogout", text=True)
    merge_tsf_files(False)

    print("Ran session 3 manually")
    subprocess.check_call("java -cp src/main/java FrontEnd validAccountList.txt
transactionSummaryFile.txt")
    merge_tsf_files(True)  # call this after every logout

    # Close the file to ensure writer is flushed to
    disk merged_tsf.close()

    print("Ran backend for day")
```

```
    subprocess.check_call("java -cp src/main/java BackEnd oldMasterAccounts.txt
mergedTransactionSummaryFile.txt")


def other_days():
    """All other days of the week should have random command input to the program

    :return: None
    """

    # Clear the merged tsf file from last session
    merged_tsf.truncate(0)
    print("--- Day " + sys.argv[1] + " ---")

    print("Ran session 1 automatically")
    set_of_commands = get_random_sequence(random.randint(1, 10))
    subprocess.run("java -cp src/main/java FrontEnd newValidAccList.txt
transactionSummaryFile.txt",
                   input=set_of_commands, text=True)
    merge_tsf_files(False)

    print("Ran session 2 automatically")
    set_of_commands = get_random_sequence(random.randint(1, 10))
    subprocess.run("java -cp src/main/java FrontEnd newValidAccList.txt
transactionSummaryFile.txt",
                   input=set_of_commands, text=True)
    merge_tsf_files(False)

    print("Ran session 3 automatically")
    set_of_commands = get_random_sequence(random.randint(1, 10))
    subprocess.run("java -cp src/main/java FrontEnd newValidAccList.txt
transactionSummaryFile.txt",
                   input=set_of_commands, text=True)
    merge_tsf_files(True)

    merged_tsf.close()

    print("Ran backend for day")
    subprocess.check_call("java -cp src/main/java BackEnd NewMasterAccountsFile.txt
mergedTransactionSummaryFile.txt")


if len(sys.argv) >= 2:
    if sys.argv[1] == "1":
        # it is the first
        day first_day()
    else:
        other_days()
else:
    print("Script malfunction, specify day")
```

Source listing of Weekly Script:

```
import subprocess

# Monday
subprocess.run("python dailyScript.py 1")

# Tuesday
subprocess.run("python dailyScript.py 2")

# Wednesday
subprocess.run("python dailyScript.py 3")

# Thursday
subprocess.run("python dailyScript.py 4")

# Friday
subprocess.run("python dailyScript.py 5")
```

Printout of the set of transaction session inputs for one run of our Daily script:

```
login
agent
createacct
1234566
armin
createacct
1234567
god
logout
login
agent
createacct
7654321
blueface
logout
login
agent
createacct
1234565
DA bank
createacct
1111111
yeezy
logout
```

A printout of the Merged Transaction Summary file from that same Daily run:

```
NEW 1234566 000 0000000 armin
NEW 1234567 000 0000000 god
EOS
NEW 7654321 000 0000000 blueface
EOS
NEW 1234565 000 0000000 DA bank
NEW 1111111 000 0000000 yeezy
EOS
EOS
```

Print out of master accounts file after day 1:

```
7654321 0 blueface
1234567 0 god
1234566 0 armin
1234565 0 DA bank
1111111 0 yeezy
```

Print out of master accounts file after day 2:

```
7654321 0 blueface
6969696 0 geazy
6955569 0 pleaseGiveUsS
1234567 0 god
1234566 0 armin
1234565 0 DA bank
1111111 0 yeezy
```

Print out of master accounts file after day 3:

```
7654321 0 blueface
6969696 0 geazy
6955569 0 pleaseGiveUsS
1234567 0 god
1234566 0 armin
1234565 0 DA bank
1111111 0 yeezy
```

Print out of master accounts file after day 4:

```
7654321 29997 blueface
6969696 0 geazy
6955569 0 pleaseGiveUsS
1234567 50000 god
1234566 0 armin
1234565 0 DA bank
1111111 0 yeezy
```

Print out of master accounts file after day 5:

```
7654321 21197 blueface
6969696 0 geazy
6955569 0 pleaseGiveUsS
1234567 50000 god
1234566 8800 armin
1234565 0 DA bank
1111111 0 yeezy
```

Integration defect report:

Problems	Solution
Missed check for frontend – monetary amounts should be between 3 and 8 digits	Added check into front end to catch false inputs
isMafValid function contained an incorrect value to check descending account number	Changing the value to eight 9s instead of seven 9s
processMergedTransactions function had index values for strings accountTo and accountFrom mixed up	Assigned accountTo and accountFrom to the correct index values
Withdraw and deposit functions in BackEnd were throwing a failed constraint log every run	Added condition in if statement so that if account name was ***, no failed constraint log would be thrown