Assignment 5

Connor Crowe - 20009994
Jordan Mack - 20005220
Michael Briggs - 20013906
Sasanka Wickramasinghe - 10192504
CMPE 327
Steven Ding
2019-11-25

For our Backend office testing, we chose to use statement coverage for the account creation transaction and decision coverage for the withdraw transactions.

Statement Coverage

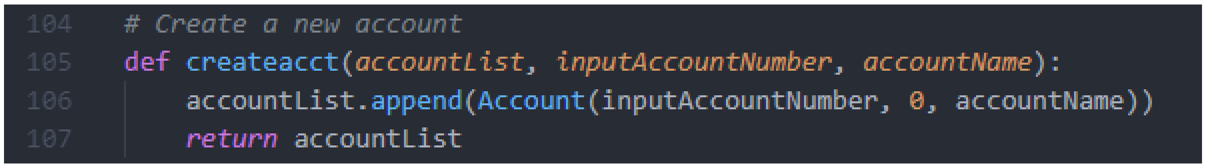
Since our createacct method is simple, the statement coverage consisted of one test case where line 106 was tested with different inputs. We chose the following inputs as shown in the table below. For the inputAccountNumber, we had to input an account number that does not already exist in the master_accounts.txt. If an invalid account number was chosen (e.g. 1234567), the statement 106 would not execute when testing. Since the frontend is able to check the validity of the account number and account name, the testing of the backend was shortened.

Statement	accountList Input	inputAccountNumber input	accountName input	Test
106	AccountList	1234555	ACCTNAME	T1
107	AccountList	1234555	ACCTNAME	

Transaction Summary Input:

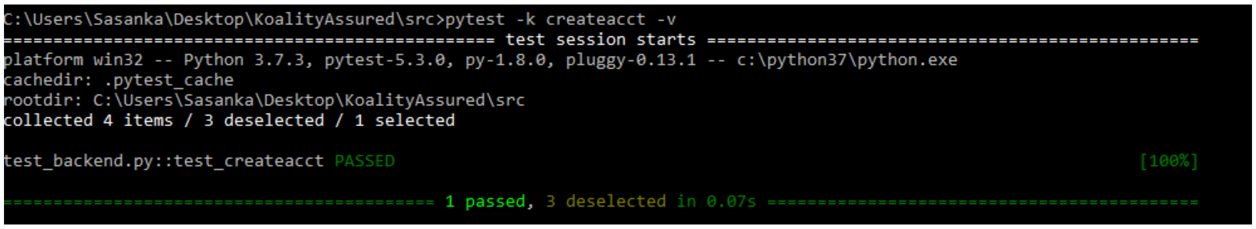
mergeT1.txt: NEW 1333333 000 0000000 INITACC

Section of code:



```
104    # Create a new account
105    def createacct(accountList, inputAccountNumber, accountName):
106        accountList.append(Account(inputAccountNumber, 0, accountName))
107        return accountList
```

Results:



```
C:\Users\Sasanka\Desktop\KoalityAssured\src>pytest -k createacct -v
=============================================== test session starts ================================================
platform win32 -- Python 3.7.3, pytest-5.3.0, py-1.8.0, pluggy-0.13.1 -- c:\python37\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Sasanka\Desktop\KoalityAssured\src
collected 4 items / 3 deselected / 1 selected

test_backend.py::test_createacct PASSED                                                                       [100%]

========================================== 1 passed, 3 deselected in 0.07s ==========================================
```

Decision Coverage

Unlike the statement coverage tests, for the decision coverage we had multiple inputs to test multiple decisions in the withdraw method. For statement 85, a valid account number would be a number that exists in the master_accounts.txt. On the other hand, an invalid account number would be a number that has not yet been added to the file (e.g. 1111111). The input amount that was to be inputted had to be less than the account's balance. If this was not the case, it meant the account had insufficient funds for that withdrawal to happen. For T4, account number

1234567 had a balance of 30,000, however the input amount was 45,000 which is greater than the balance. This resulted in the test to fail.

Decision	**accountList Input**	**inputAccountNumber input**	**inputAmount input**	**Test**
85: true	AccountList	1234567	100	T1
85: false	AccountList	1111111	100	T2
86: true	AccountList	1234567	100	T3
86: false	AccountList	1234567	45000	T4

Transaction Summary Input:

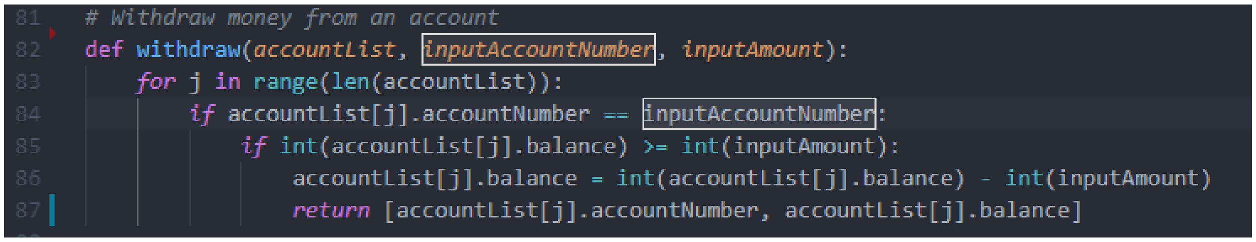
mergeT1.txt: WDR 1234567 100 0000000 ***

mergeT1.txt: WDR 1111111 100 0000000 ***

mergeT1.txt: WDR 1234567 100 0000000 ***

mergeT1.txt: WDR 1234567 45000 0000000 ***

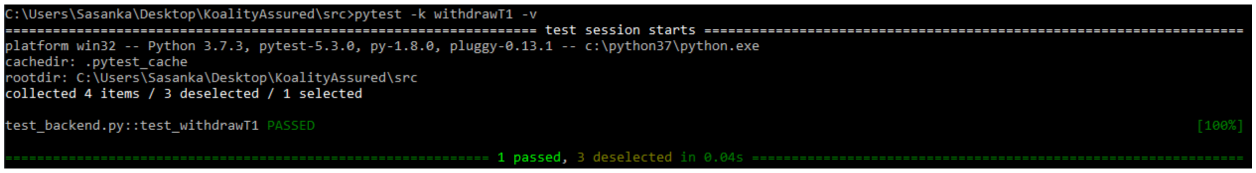
Section of code:



```
81  # Withdraw money from an account
82  def withdraw(accountList, inputAccountNumber, inputAmount):
83      for j in range(len(accountList)):
84          if accountList[j].accountNumber == inputAccountNumber:
85              if int(accountList[j].balance) >= int(inputAmount):
86                  accountList[j].balance = int(accountList[j].balance) - int(inputAmount)
87                  return [accountList[j].accountNumber, accountList[j].balance]
```

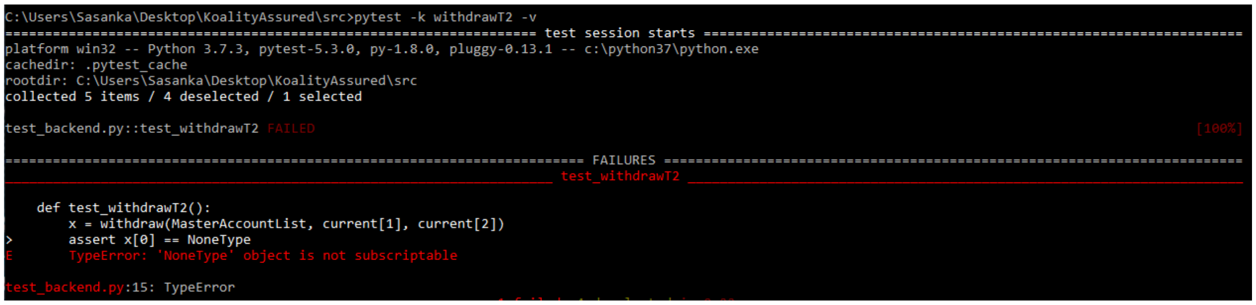
Results:

Result for test_withdrawT1()



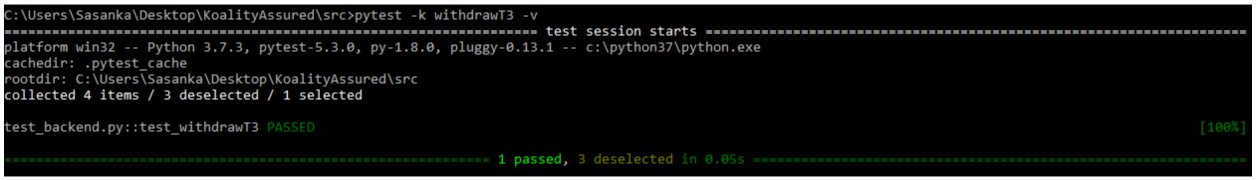
```
C:\Users\Sasanka\Desktop\KoalityAssured\src>pytest -k withdrawT1 -v
=================================================================== test session starts ====================================================================
platform win32 -- Python 3.7.3, pytest-5.3.0, py-1.8.0, pluggy-0.13.1 -- c:\python37\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Sasanka\Desktop\KoalityAssured\src
collected 4 items / 3 deselected / 1 selected

test_backend.py::test_withdrawT1 PASSED                                                                                                               [100%]

============================================================= 1 passed, 3 deselected in 0.04s =============================================================
```

Result for test_withdrawT2()



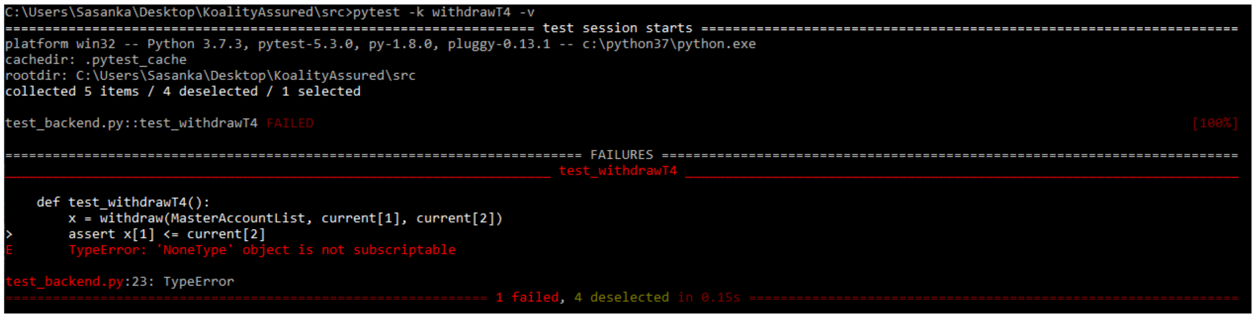
```
C:\Users\Sasanka\Desktop\KoalityAssured\src>pytest -k withdrawT2 -v
=================================================================== test session starts ===================================================================
platform win32 -- Python 3.7.3, pytest-5.3.0, py-1.8.0, pluggy-0.13.1 -- c:\python37\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Sasanka\Desktop\KoalityAssured\src
collected 5 items / 4 deselected / 1 selected

test_backend.py::test_withdrawT2 FAILED                                                                                                              [100%]

======================================================================== FAILURES =========================================================================
______________________________________________________________________ test_withdrawT2 ____________________________________________________________________

    def test_withdrawT2():
        x = withdraw(MasterAccountList, current[1], current[2])
>       assert x[0] == NoneType
E       TypeError: 'NoneType' object is not subscriptable

test_backend.py:15: TypeError
```

Result for test_withdrawT3()



```
C:\Users\Sasanka\Desktop\KoalityAssured\src>pytest -k withdrawT3 -v
=================================================================== test session starts ===================================================================
platform win32 -- Python 3.7.3, pytest-5.3.0, py-1.8.0, pluggy-0.13.1 -- c:\python37\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Sasanka\Desktop\KoalityAssured\src
collected 4 items / 3 deselected / 1 selected

test_backend.py::test_withdrawT3 PASSED                                                                                                              [100%]

============================================================ 1 passed, 3 deselected in 0.05s ============================================================
```

Result for test_withdrawT4()



```
C:\Users\Sasanka\Desktop\KoalityAssured\src>pytest -k withdrawT4 -v
=================================================================== test session starts ===================================================================
platform win32 -- Python 3.7.3, pytest-5.3.0, py-1.8.0, pluggy-0.13.1 -- c:\python37\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\Sasanka\Desktop\KoalityAssured\src
collected 5 items / 4 deselected / 1 selected

test_backend.py::test_withdrawT4 FAILED                                                                                                              [100%]

======================================================================== FAILURES =========================================================================
______________________________________________________________________ test_withdrawT4 ____________________________________________________________________

    def test_withdrawT4():
        x = withdraw(MasterAccountList, current[1], current[2])
>       assert x[1] <= current[2]
E       TypeError: 'NoneType' object is not subscriptable

test_backend.py:23: TypeError
============================================================ 1 failed, 4 deselected in 0.15s ============================================================
```

Team Contribution

Member	Time Spent (hours)	Tasks
Connor Crowe	7	Worked on report, and created test cases
Jordan Mack	7	Worked on report, and created test case
Michael Briggs	7	Worked on report, and testing test cases

Sasanka Wickramasinghe	7	Worked on report, and testing test cases