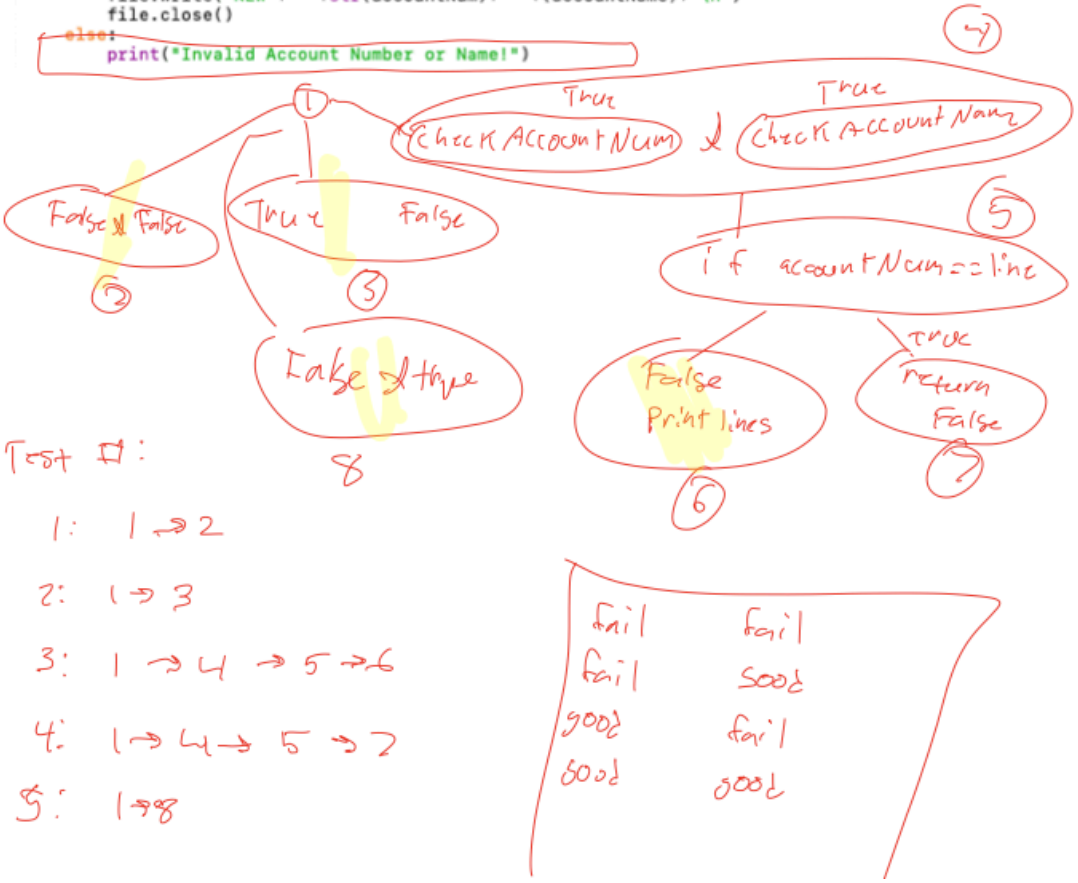
327 Assignment 5

Create account:
1.) We have chosen to do the path coverage white box testing plan for this method.
2.)

```
def createacct_action(accountNum, accountName):
    if(checkAccountNum(accountNum)==True and checkAccountName(accountName)==True):
        valid = open("validAccountsList.txt", "r")
        for line in valid:
            if(accountNum==line):
                print("Account number already being used by another user try again.")
                return False

        #save account num and name to transaction summary file
        print("Created account!")
        print("Account Info:")
        print("Account Number: " + str(accountNum))
        print("Account Name: " + str(accountName))
        today = date.today()
        fileName = ("DailyTransactions"+str(today)+".txt")
        file = open(fileName,"a+")
        file.write("NEW"+" "+str(accountNum)+" "+(accountName)+"\n")
        file.close()
    else:
        print("Invalid Account Number or Name!")
```



As shown above this is how we broke down the paths. At every if statement it was either true or false and we made a test case for every possible outcome.

Path	AccountNum Valid	AccountName valid	accountNum unique
P1	TRUE	TRUE	TRUE
P2	TRUE	FALSE	FALSE
P3	FALSE	TRUE	FALSE
P4	FALSE	FALSE	FALSE
P5	TRUE	TRUE	FALSE

3.)

Test case	Input	Output
P1	['NEW', '1311111', '10000', '1311111', 'Micael']	`Created account!`
P2	['NEW', '14567', '100000', '1234567', 'Michael']	`Invalid Account Number or Name!`
P3	['NEW', '1567', '100000', '1234567', 'Doug']	`Invalid Account Number or Name!`
P4	['NEW', '1111111', '10000', '1111111', 'Michael']	`Created account!`
P5	['NEW', '1234567', '10000', '1234567', 'Michael']	`Created account!`

4.) All tests passed

Withdrawal account:

1.)

For this function we did statement coverage.

```
def withdraw_action(master, transaction):
    if(!transaction[0] == "WDR"):
            print("This is not the proper withdrawl code")
    if(!len(transaction[1]) == 7):
            print("This is not the proper user ID number")
    if(!transaction[0] == 0):
            print("This is not the proper user ID number")
    if(!transaction[2] == 0):
            print("This is not enough money to withdrawl")
    if(!len(transaction[3]) == 7):
            print("This is not the proper user ID number")
    if(!transaction[3] == 0):
            print("This is not the proper user ID number")
    person_id = transaction[1]
    amount = transaction[2]
    person_name = transaction[4]
    for account in master:
        if master[account][0] == person_id and master[account][2] == person_name:
            master[account][1] = int(master[account][1]) - int(amount)
```

Statement coverage:

make 7 tests to run each of the # lines of code @ least 1

2.)

Test Number	Input
P1	['WDR', '234567', '100000', '1234567', 'Michael']
P2	['WDR', '0234567', '100000', '1234567', 'Michael']
P3	['WDR', '1234567', '0', '1234567', 'Michael']
P4	['WDR', '1234567', '10000', '123456', 'Michael']
P5	['WDR', '1234567', '10000', '0234567', 'Michael']
P6	['WDR', '1111111', '10000', '1111111', 'Michael']
P7	['WDR', '11111', '10000', '1111111', 'Michael']

3.)

Test Number	Output
P1	This is not the proper user ID number
P2	This is not the proper user ID number
P3	This is not the proper user ID number
P4	This is not the proper user ID number

Test Number	Output
P5	`This is not the proper user ID number`
P6	`This is not the proper user ID number`
P7	`This is not the proper user ID number`

4.)

This all worked as it should have except for p7 which did not output correctly. This test failed even though it should have printed the statement “withdrew 100000”.