Unit Testing - Using *doctest*

In your **doctest-lab** folder, look at **BankAccount.py**. This program contains a class definition of a typical Bank Account, like we've seen already. The program contains comments describing what to write doctests for. For example, in the *balance property:*

```
"""
Test if invalid balance for setter (-1000) prints diagnostic
 **** CODE STUFF HERE ****
'Value -1000 invalid for balance field; no change made'
"""
```

Write the doctest statements that will apply a test to produce the shown output.

There are two additional comment blocks: one in *make_deposit* and the other in *transfer*. Complete these comment blocks with the appropriate doctest statements.

You may want to **run various commands in the shell** first to get the **exact** outputs.

Next page has expected output.

Expected Output

```
D:\...>python solution.py -v

Trying:
   acct_4_test = BankAccount(100, "Joe Smith", 1_000)
Expecting nothing

ok

Trying:
   acct_4_test.make_deposit(-1000)
Expecting:
   Value -1000 invalid for deposit amount; no change made

ok

Trying:
   existing = acct_4_test.balance
Expecting nothing

ok

Trying:
   acct_4_test.make_deposit(200)
Expecting nothing

ok

Trying:
   new = acct_4_test.balance
Expecting nothing

ok

Trying:
   new == existing + 200
Expecting:
   True

ok
```

```
Trying:
   acct_4_test = BankAccount(100, "Joe Smith", 1_000)
Expecting nothing

ok

Trying:
    diff_acct = BankAccount(100, "Jane Doe", 1_000)
Expecting nothing

ok

Trying:
    diff_acct.transfer(300, acct_4_test)
Expecting:
    Account holders not the same; no changes made

ok

6 items had no tests:
    __main__
    __main__.BankAccount
    __main__.BankAccount.__init__
    __main__.BankAccount.balance
    __main__.BankAccount.make_withdrawal
    __main__.BankAccount.print_acct_info
2 items passed all tests:
   6 tests in __main__.BankAccount.make_deposit
   3 tests in __main__.BankAccount.transfer
9 tests in 8 items.
9 passed and 0 failed.
Test passed.
```