stock_broker testplan

WO1 Clayton E. Williams

October 2023

1 Purpose

stock_broker is a command line utility tool that, through the use of command-line options, allows a broker to manage customers, their accounts, and the various functions associated with them, such as depositing and withdrawing funds, and buying and selling stocks.

The purpose of this test plan is to provide a process of testing the program against a variety of command-line inputs to gain reasonable assurance that the program exhibits desired behavior and does not crash on unexpected or bad input.

2 Components

The test coverage of stock_broker does not contain any manual testing and is encompassed entirely within 2 test modules

2.1 Test Modules

- test_customer.py
- test_account.py

2.2 Test Cases

- test_customer
  - test_fields
    This test case tests the Customer class constructor. This tests against an object instantiation that the values passed are the same returned from the class getters. Additionally, a second object is created, testing that the unique ID increments and is different from object to object.
- test_account

- test_acct_attributes This test case tests the Account class constructer. This tests against an object instantiation that the values passed are the same returned from the class getters.
- test_acct_deposit With an instantiated Account object, this test case tests the deposit function with valid and invalid input, asserting that the balance is reflected appropriately.
- test_acct_withdraw With an instantiated Account object of balance 100, this test case withdraws 50, asserting the balance is 50. It then attempts to withdraw 100, which is out of bounds and asserts the value remains unchanged.
- test_holding_attributes This test case tests the Holding class constructor. This tests against an object instantiation that the values passed are the same as those returned from the class getters.
- test_holding_sell With a Holding object of 1 share, this test case calls the sell_shares() function with 1 share, asserting the current shares is now 0.
- test_holding_buy With a Holding object of 1 share, this test case calls the buy_shares() function with 2 shares, and asserts the current shares is now 3.
- test_transaction_attributes this test case tests the Transaction class constructor. This tests against an object instantiation that the values passed are the same as those returned from the class getters.

3 Running Test Suite

Running the automated test suite can be done using Python's unittest module:

```
$python3 -m unittest discover -s test
```

Which results in the following:

```
Ran 8 tests in 0.001s

OK
```