**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №5

«**Модульное тестирование в Python.**»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б  Казицин Алексей |  | преподаватель кафедры  Гапанюк Ю.Е. |
|  |  |  |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2022 г.

**Цель лабораторной работы:** изучение возможностей модульного тестирования в языке Python.

**Требования к отчету:**

Отчет по лабораторной работе должен содержать:

1. титульный лист;
2. описание задания;
3. текст программы;
4. экранные формы с примерами выполнения программы.

**Задание:**

1. Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.
2. Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.
3. Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:
   * TDD - фреймворк (не менее 3 тестов).
   * BDD - фреймворк (не менее 3 тестов).
   * Создание Mock-объектов (необязательное дополнительное задание).

Текст программы:

Test\_TDD.py

import unittest  
  
import sort  
  
  
class test\_sort(unittest.TestCase):  
 def test\_sort\_1\_1(self):  
 self.assertEqual(sort.sort\_1([3, -4, 5, 0, 1]), [5, -4, 3, 1, 0])  
  
 def test\_sort\_1\_2(self):  
 self.assertEqual(sort.sort\_1([3, -4, 4, 5, 0, 1, -1, 17]), [17, 5, -4, 4, 3, 1, -1, 0])  
  
 def test\_sort\_1\_3(self):  
 self.assertEqual(sort.sort\_1([0, -100, 100, 67, -67, 67, 99, 15, 16, -15]),  
 [-100, 100, 99, 67, -67, 67, 16, 15, -15, 0])  
  
 def test\_sort\_2\_1(self):  
 self.assertEqual(sort.sort\_2([3, -4, 5, 0, 1]), [5, -4, 3, 1, 0])  
  
 def test\_sort\_2\_2(self):  
 self.assertEqual(sort.sort\_2([3, -4, 4, 5, 0, 1, -1, 17]), [17, 5, -4, 4, 3, 1, -1, 0])  
  
 def test\_sort\_2\_3(self):  
 self.assertEqual(sort.sort\_2([0, -100, 100, 67, -67, 67, 99, 15, 16, -15]),  
 [-100, 100, 99, 67, -67, 67, 16, 15, -15, 0])  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 unittest.main()

Sort.py

data\_1 = [4, -30, 100, -100, 123, 1, 0, -1, -4]  
  
  
def sort\_1(data):  
 result = sorted(data, key=abs, reverse=True)  
 return result  
  
  
def sort\_2(data):  
 result\_with\_lambda = sorted(data, key=lambda x: abs(x), reverse=True)  
 return result\_with\_lambda  
  
  
def main\_s():  
 result = sort\_1(data\_1)  
 print(result)  
  
 result\_with\_lambda = sort\_2(data\_1)  
 print(result\_with\_lambda)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main\_s()

Test1.py

from behave import \*  
import sort  
  
  
@given('the list is [3, -4, 5, 0, 1]')  
def step\_impl(context):  
 context.gdata = [3, -4, 5, 0, 1]  
  
  
@when('the list is sorted')  
def step\_impl(context):  
 context.gdata = sort.sort\_1(context.gdata)  
  
  
@then('the new list is [5, -4, 3, 1, 0]')  
def step\_impl(context):  
 assert context.gdata == [5, -4, 3, 1, 0]

Test2.py

from behave import \*  
import sort  
  
  
@given('the list is [3, -4, 4, 5, 0, 1, -1, 17]')  
def step\_impl(context):  
 context.gdata = [3, -4, 4, 5, 0, 1, -1, 17]  
  
  
@when('the list is sorted2')  
def step\_impl(context):  
 context.gdata = sort.sort\_1(context.gdata)  
  
  
@then('the new list is [17, 5, -4, 4, 3, 1, -1, 0]')  
def step\_impl(context):  
 assert context.gdata == [17, 5, -4, 4, 3, 1, -1, 0]

Test3.py

from behave import \*  
import sort  
  
  
@given('the list is [0, -100, 100, 67, -67, 67, 99, 15, 16, -15]')  
def step\_impl(context):  
 context.gdata = [0, -100, 100, 67, -67, 67, 99, 15, 16, -15]  
  
  
@when('the list is sorted3')  
def step\_impl(context):  
 context.gdata = sort.sort\_1(context.gdata)  
  
  
@then('the new list is [-100, 100, 99, 67, -67, 67, 16, 15, -15, 0]')  
def step\_impl(context):  
 assert context.gdata == [-100, 100, 99, 67, -67, 67, 16, 15, -15, 0]

Tutorial.feature

Feature: Abs\_sorting  
  
 Scenario: Sort Abs  
 Given the list is [3, -4, 5, 0, 1]  
 When the list is sorted  
 Then the new list is [5, -4, 3, 1, 0]  
  
 Scenario: Sort Abs2  
 Given the list is [3, -4, 4, 5, 0, 1, -1, 17]  
 When the list is sorted2  
 Then the new list is [17, 5, -4, 4, 3, 1, -1, 0]  
  
 Scenario: Sort Abs3  
 Given the list is [0, -100, 100, 67, -67, 67, 99, 15, 16, -15]  
 When the list is sorted3  
 Then the new list is [-100, 100, 99, 67, -67, 67, 16, 15, -15, 0]

Запуск программы:



