Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика, искусственный интеллект и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе «Модульное тестирование в Python»

Выполнила:

студентка группы ИУ5-34Б:

Теряева Ксения Владимировна
14.12.2022г

Проверил:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Москва, 2022 г

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:

Выбранный для тестирования фрагмент 3-4 лабораторной:

```
#4
def sort_with_lambda(arr):
    return sorted(arr, key = lambda x: -abs(x))
mas1 = [4, -30, 100, -100, 123, 1, 0, -1, -4]
if __name__ == '__main__':
    mas3 = sorted(mas1,key = lambda x: -abs(x))
    print(mas3)

    mas2 = sorted(mas1, key = abs, reverse = True)
    print(mas2)
```

TDD тест:

```
import unittest
from task4 import sort_with_lambda

class Test_task(unittest.TestCase):
    def test_empty(self):
        self.assertEqual(sort_with_lambda([]),[])
    def test_sort(self):
        mas1 = [4, -30, 100, -100, 123, 1, 0, -1, -4]
        res = [123, 100, -100, -30, 4, -4, 1, -1, 0]
        self.assertEqual(sort_with_lambda(mas1),res)
    def test_first(self): #проверяет что мы не изменили начальный массив
        mas1 = [4, -30, 100, -100, 123, 1, 0, -1, -4]
        sort_with_lambda(mas1)
        self.assertEqual(mas1, [4, -30, 100, -100, 123, 1, 0, -1, -4])


if __name__ == '__main__':
    unittest.main()
```

BDD тест:

```
Feature: Sort
    A function that sorts an array modulo in reverse order

    Scenario: Empty array
        Given Empty array
        When We call a function from it
        Then It should still be empty

    Scenario: Small array
        Given Small array
        When We call a function from small array
        Then It should return new array which is sorted modulo in reverse order

    Scenario: Big array
        Given Big array
        When We call a function from big array
        Then It should return new big array which is sorted modulo in reverse order
```

```
from pytest_bdd import scenario, given, when, then
from task4 import sort_with_lambda
import random as rd

@scenario('bdd_sort.feature','Empty array')
def test_empty_array():
        pass

@given('Empty array',target_fixture= "arr")
def give_empty_arr():
        return []

@when("We call a function from it",target_fixture='sort_arr')
def call_func_empty(arr):
        return sort_with_lambda(arr)

@then('It should still be empty')
def eq_empty(sort_arr):
        assert ([] == sort_arr)


@scenario('bdd_sort.feature','Small array')
def test_small_array():
        pass

@given('Small array',target_fixture= "arr")
def give_small_arr():
        arr = list(range(10))
        rd.shuffle(arr)
        return arr

@when("We call a function from small array",target_fixture='sort_arr')
def call_func_small(arr):
        return sort_with_lambda(arr)
```

```
@then('It should return new array which is sorted modulo in reverse order')
def eq_small(sort_arr):
        assert (list(range(10)) == sort_arr[::-1])


@scenario('bdd_sort.feature','Big array')
def test_big_array():
        pass

@given('Big array',target_fixture= "arr")
def give_big_arr():
        return [790, 339, 313, -441, 773, 246, 737, 818, 248, -412, -514,
                        -361, -973, -686, 763, 777, -827, -5, -37, -23, 553, -92,
                        145, -620, 851, -429, 564, 986, 26, 464, -505, -366, 519, -428,
                        -676, -776, -493, -784, -377, 336, 786, -7, -243, -699, -825, -893,
                        -291, 76, -14, -535]

@when("We call a function from big array",target_fixture='sort_arr')
def call_func_big(arr):
        return sort_with_lambda(arr)

@then('It should return new big array which is sorted modulo in reverse order')
def eq_big(sort_arr):
        ans = [986, -973, -893, 851, -827, -825, 818, 790, 786, -784, 777,
                  -776, 773, 763, 737, -699, -686, -676, -620, 564, 553, -535, 519, -514,
                  -505, -493, 464, -441, -429, -428, -412, -377, -366, -361, 339, 336, 313,
                  -291, 248, 246, -243, 145, -92, 76, -37, 26, -23, -14, -7, -5]
        assert (ans == sort_arr)
```

Результат работы программы:

```
= RESTART: C:\Users\Ксения\Desktop\3ий сем\proger\lab4\test_tdd.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.010s
```