Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «БКИТ»
Отчет по лабораторной работе №5
«Модульное тестирование в Python»

Выполнил:
студент группы ИУ5-35Б
Большаков Георгий
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Нардид А.Н.
Подпись и дата:

Москва, 2022 г.

Описание задания

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

Файлы для BDD-тестирования

scenario.feature

```
Feature: Testing
Scenario: 2 roots
Given nums
When equation is solved
Then roots are
```

Lab1.py

```
import random
import sys
import math

def get_num(index, promt):
    try:
        num_str = sys.argv[index]
    except:
        print(promt)
        num_str = input()

    if not num_str.isalpha():
        num = float(num_str)
    else:
        print("Коэффициент задан некорректно. Он будет задан рандомно от 1
до 10")
        num = random.randint(1, 10)
        print(num)
    return num

def get_roots(a, b, c):
    result = []
    if a == 0:
        print("коэффициент A не может быть равне нулю. он будет задан
рандомно от 1 до 10")
        a = random.randint(1, 10)
        print(a)
    D = b * b - 4 * a * c
    if D == 0.0:
        t = -b / (2.0 * a)
        x1 = - math.sqrt(t)
        x2 = math.sqrt(t)
        result.append(x1)
        if x1 != 0:
            result.append(x2)
    elif D > 0.0:
        t1 = (- b + math.sqrt(D)) / (2.0 * a)
        if t1 > 0.0:
            x1_1 = math.sqrt(t1)
            x1_2 = - math.sqrt(t1)
```

```
            result.append(x1_1)
            result.append(x1_2)

        t2 = (- b - math.sqrt(D)) / (2.0 * a)
        if t2 > 0.0:
            x2_1 = math.sqrt(t2)
            x2_2 = - math.sqrt(t2)
            result.append(x2_1)
            result.append(x2_2)

    return result

def main():
    a = get_num(1, "Введите коэффициент A:")
    b = get_num(2, "Введите коэффициент B:")
    c = get_num(3, "Введите коэффициент C:")

    roots = get_roots(a, b, c)
    len_roots = len(roots)
    if len_roots == 0:
        print("Нет корней")
    elif len_roots == 1:
        print("Один корень: {}".format(roots[0]))
    elif len_roots == 2:
        print("Два корня: {} и {}".format(roots[0], roots[1]))
    elif len_roots == 4:
        print("Четыре корня: {}; {}; {}; {};".format(roots[0], roots[1],
roots[2], roots[3]))

    # если сценарий запущен из командной строки
if __name__ == "__main__":
    main()
```

test.py

```
import unittest
import pytest
from pytest_bdd import feature, scenario, given, when, then
from main import get_roots
from unittest import TestCase


class GetNumCoef(TestCase):
    def test1(self):
        self.assertEqual(get_roots(3, 7, -10), [1.0, -1.0])
    def test2(self):
        self.assertEqual(get_roots(1, 1, 1), [])
    def test3(self):
        self.assertEqual(get_roots(8, -6, 1), [0.7071067811865476, -
0.7071067811865476, 0.5, -0.5])


@scenario("scenarios.feature", "2 roots")
def test1():
```

```
    print("\nScenario: 2 roots")

@given("nums")
def test1():
    print("\nnums: {[3, 7, -10]}")

@when('equasion is solved')
def test2():
    print('\nequasion is solved')

@then('roots are')
def test3():
    print('\nroots are:',1.0, -1.0)
    assert get_roots(3, 7, -10) == [1.0, -1.0]

def main():
    unittest.main()
```

Экранные формы

```
============================= test session starts ==============================
collecting ... collected 6 items

test.py::GetNumCoef::test1 PASSED                                        [ 16%]
test.py::GetNumCoef::test2 PASSED                                        [ 33%]
test.py::GetNumCoef::test3 PASSED                                        [ 50%]
test.py::test1 PASSED                                                    [ 66%]
nums: {[3, 7, -10]}

test.py::test2 PASSED                                                    [ 83%]
equasion is solved

test.py::test3 PASSED                                                    [100%]
roots are: 1.0 -1.0


============================== 6 passed in 0.03s ===============================
```