Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

ОТЧЁТ ПО

Лабораторной работе №5

«Модульное тестирование в Python.»

Выполнил:

студент группы ИУ5-35Б

Нгуен Зуи Лам

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Нардид А.Н.

Подпись и дата:

Москва

2022

Задания

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.

Файл main.py

```
import math

def get_roots(a, b, c):
    result = []
    D = b * b - 4 * a * c
    if a == 0.0 and b != 0.0:
        if -c / b >= 0.0:
            root = math.sqrt(-c/b)
            if root != 0.0:
                result.append(-root)
                result.append(root)
            else:
                result.append(abs(root))
    elif a == 0.0 and b == 0.0:
        if c == 0.0:
            print('Бесконечное множество корней')
            exit(1)
        else:
            print('Нет корней')
            exit(1)
    elif D == 0.0:
        if -b / (2.0 * a) > 0.0:
            root = math.sqrt(-b / (2.0 * a))
            if root != 0.0:
                result.append(root)
                result.append(-root)
            else:
                result.append(abs(root))
    elif D > 0.0:
        if (-b + math.sqrt(D)) / (2.0 * a) > 0.0:
            root1 = math.sqrt((-b + math.sqrt(D)) / (2.0 * a))
            if root1 != 0.0:
                result.append(root1)
                result.append(-root1)
            else:
                result.append(abs(root1))
        if (-b - math.sqrt(D)) / (2.0 * a) > 0.0:
```

```
            root2 = math.sqrt((-b - math.sqrt(D)) / (2.0 * a))
            if root2 != 0.0:
                result.append(root2)
                result.append(-root2)
            else:
                result.append(abs(root2))
    return result
```

Файл testing.py

```
from math import *
from main import *
import unittest
from behave import Given, Then, When

class Test_get_rootsTTD(unittest.TestCase):
    def test1(self):
        self.assertEqual(get_roots(1, -4, 4), [sqrt(2), -sqrt(2)])

    def test2(self):
        self.assertEqual(get_roots(9, -10, 1), [1, -1, 1/3, -1/3])

    def test3(self):
        self.assertEqual(get roots(0, 1, -4), [-2, 2])

    def test4(self):
        self.assertEqual(get_roots(1, 11, 10), [])


"Data for equation"
def test1():
    print("Data for equation")
    print(f"Data: {[9, -10, 1]}")
    print(f"Data: {[1, -4, 4]}")


@When("I want to solve the equation")
def test2():
    print("I want to solve the equation")


@Then("I get roots")
def test3():
    print("I get roots")
    assert get_roots(9, -10, 1) == [1, -1, 1/3, -1/3]
    assert get_roots(1, -4, 4) == [sqrt(2), -sqrt(2)]

def main():
    unittest.main()


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

Файл f.feature

```
Feature: Testing

  Scenario: 4 roots
    Given Data for equation
```

```
    When I want to solve the equation

    Then I get roots
```

Результат выполнения.

```
D:\PyCharm\lab4\venv\Scripts\python.exe "D:/PyCharm/PyCharm Community Edition 2022.2.2/plugins/python-ce/helpers/pycharm/_jb_pytest_runner
 .py" --path D:\PyCharm\lab4\test\testing.py
Testing started at 23:36 ...
Launching pytest with arguments D:\PyCharm\lab4\test\testing.py --no-header --no-summary -q in D:\PyCharm\lab4\test

============================ test session starts ============================
collecting ... collected 7 items

testing.py::Test_get_rootsTTD::test1 <- Testing.py PASSED               [ 14%]
testing.py::Test_get_rootsTTD::test2 <- Testing.py PASSED               [ 28%]
testing.py::Test_get_rootsTTD::test3 <- Testing.py PASSED               [ 42%]
testing.py::Test_get_rootsTTD::test4 <- Testing.py PASSED               [ 57%]
testing.py::test1 <- Testing.py PASSED                                  [ 71%]Data for equation
Data: [9, -10, 1]
Data: [1, -4, 4]

testing.py::test2 <- Testing.py PASSED                                  [ 85%]I want to solve the equation

testing.py::test3 <- Testing.py PASSED                                  [100%]I get roots


============================ 7 passed in 0.09s =============================

Process finished with exit code 0
```