**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по Лабораторной работе №5

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б: |  | преподаватель каф. ИУ5 |
| Лапотко Ирина Евгеньевна |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2022 г.

**Постановка задачи:**

Выберите любой фрагмент кода из лабораторных работ 1 или 2 или 3-4.

Модифицируйте код таким образом, чтобы он был пригоден для модульного тестирования.

Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:

TDD - фреймворк (не менее 3 тестов).

BDD - фреймворк (не менее 3 тестов).

**Текст программы:**

**Func.py**

import math

def get\_roots(a, b, c):

    result = []

    D = b\*b - 4\*a\*c

    if D == 0.0:

        y = -b / (2.0\*a)

        if y >= 0:

            root\_1 = math.sqrt(y)

            root\_2 = -math.sqrt(y)

            result.append(root\_1)

            result.append(root\_2)

    elif D > 0.0:

        sqD = math.sqrt(D)

        y1 = (-b + sqD) / (2.0\*a)

        y2 = (-b - sqD) / (2.0\*a)

        if (y1 >= 0):

            root1\_1 = math.sqrt(y1)

            root1\_2 = -math.sqrt(y1)

            result.append(root1\_1)

            result.append(root1\_2)

        if (y2 >= 0):

            root2\_1 = math.sqrt(y2)

            root2\_2 = -math.sqrt(y2)

            result.append(root2\_1)

            result.append(root2\_2)

    return result

**Steps.py**

import unittest

from func import get\_roots

from behave import given, when, then, step

class func\_test(unittest.TestCase):

    def test1(self):

        res = get\_roots(1, -5, 4)

        self.assertEqual(res, [2, -2, 1, -1])

    def test2(self):

        res = get\_roots(1, -5, -36)

        self.assertEqual(res, [3, -3])

    def test3(self):

        res = get\_roots(1, 10, 9)

        self.assertEqual(res, [])

@given("we have coef")

def step\_impl(context):

    pass

@when('we solve the equation with [1, -5, 4]')

def step\_impl(context):

    print('I solve the equation')

@then('we get roots')

def step\_impl(context):

    print('I get roots')

    assert get\_roots(1, -5, 4) == [2, -2, 1, -1]

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()

**4roots.feature**

Feature: Solution of equation

Scenario: Get 4 roots

Given we have coef

When we solve the equation with [1, -5, 4]

Then we get roots