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 ИУ5

«Системы обработки информации и управления»

Отчёт по лабораторной работе №5

Выполнила:
Студент группы ИУ5-31Б
Сигал Д.Э.

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:

Программа первой лабораторной(переделанная):

```
import sys
import math

def get_coef(index, prompt):
    '''
    Читаем коэффициент из командной строки или вводим с клавиатуры
    Args:
        index (int): Номер параметра в командной строке
        prompt (str): Приглашение для ввода коэффицента
    Returns:
        float: Коэффициент квадратного уравнения
    '''
    try:
        # Пробуем прочитать коэффициент из командной строки
        coef_str = sys.argv[index]
        while 1:
          try:
            coef=float(coef_str)
          except:
             print('Неверный ввод')
          else:
            break
    except:
        # Вводим с клавиатуры
        while 1:
          try:
            print(prompt)
            coef = float(input())
          except:
```

```
            print('Неверный ввод')
        else:
            break
    return coef
def get_roots(a, b, c):
    '''
    Вычисление корней квадратного уравнения
    Args:
        a (float): коэффициент A
        b (float): коэффициент B
        c (float): коэффициент C
    Returns:
        list[float]: Список корней
    '''
    result = []
    D = b*b - 4*a*c
    if a==0 and b==0:
      return result
    elif a==0:
      root=(-c/b)
      if root == 0:
          result.append(0)
      if root>0:
        result.append(-(root**0.5))
        result.append(root**0.5)
    elif D == 0.0:
        root = (-b / (2.0*a))
        if root == 0:
          result.append(0)
        elif root>0:
          result.append(-(root**0.5))
          result.append(root**0.5)
    elif D > 0.0:
        sqD = math.sqrt(D)
        root1 = (-b + sqD) / (2.0*a)
        if root1 == 0:
          result.append(0)
        if root1>0:
          result.append(-(root1**0.5))
          result.append(root1**0.5)
        root2 = (-b - sqD) / (2.0*a)
        if root2 == 0:
          result.append(0)
        if root2>0:
          result.append(-(root2**0.5))
          result.append(root2**0.5)

    return result
```

```
def main():
    '''
    Основная функция
    '''
    a = get_coef(1, 'Введите коэффициент A:')
    b = get_coef(2, 'Введите коэффициент B:')
    c = get_coef(3, 'Введите коэффициент C:')
    # Вычисление корней
    roots = get_roots(a,b,c)
    # Вывод корней
    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 == 3:
        print('Три корня: {}, {} и {}'.format(roots[0],
roots[1],roots[2]))
    elif len_roots == 4:
        print('Четыре корня: {}, {}, {} и {}'.format(roots[0],
roots[1],roots[2],roots[3]))


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

# Пример запуска
# qr.py 1 0 -4
```

Tdd проверка:

```
import unittest
from lab1re import get_roots
class TestLab1(unittest.TestCase):
  def test_noroot(self):
    coef = get_roots(1,11,10)
    self.assertEqual(len(coef),0)

  def test_oneroot(self):
    coef = get_roots(10,0,0)
    self.assertEqual(len(coef),1)
    self.assertEqual(coef, [0])
  def test_tworoot(self):
```

```
    coef = get_roots(1,-2,-8)
    self.assertEqual(len(coef),2)
    self.assertEqual(coef, [-2,2])
  def test_threeroot(self):
    coef = get_roots(-4,16,0)
    self.assertEqual(len(coef),3)
    self.assertEqual(coef, [0,-2,2])
  def test_fourroot(self):
    coef = get_roots(1,-10,9)
    self.assertEqual(len(coef),4)
    self.assertEqual(coef, [-3,3,-1,1])
if __name__ == '__main__':
    unittest.main()
```

экранные формы с примерами выполнения программы:

```
Ran 5 tests in 0.001s

OK
```

Bdd проверка:

```
from behave import *
import sys
sys.path.append("../LAB5")
from lab1re import get_roots


@given(u'{given_a} coef a, coef b is {given_b} and c is {given_c}')
def step_impl(context, given_a, given_b, given_c):
    global a
    global b
    global c
    a = int(given_a)
    b = int(given_b)
    c = int(given_c)
    return True


@When("starting function")
def step_impl(context):
    global result
    result = get_roots(a, b, c)
    # return True
    if type(a) == int:
        return True

@Then("we shoud see {given_result}")
def step_impl(context, given_result):
    try:
```

```
        assert(result == given_result)
        return True
    except:
        return False
```

feature:

```
Feature: testing roots

    Scenario Outline: multiple roots roots
        Given <a> coef a, coef b is <b> and c is <c>
        When starting function
        Then we shoud see <result>

        Examples:
            | a  | b   | c  | result          |
            | 1  | 10  | 11 | "[]"            |
            | 10 | 0   | 0  | "[0]"           |
            | 1  | -2  | -8 | "[-2, 2]"       |
            | 4  | 16  | 0  | "[-2, 0, 2]"    |
            | 1  | -10 | 9  | "[-3, 1, 1, 3]"|
```

экранные формы с примерами выполнения программы:

```
Feature: testing roots # features/test.feature:1

  Scenario Outline: multiple roots roots -- @1.1   # features/test.feature:10
    Given 1 coef a, coef b is 10 and c is 11       # features/steps/test.py:7
    When starting function                         # features/steps/test.py:18
    Then we shoud see "[]"                         # features/steps/test.py:26

  Scenario Outline: multiple roots roots -- @1.2   # features/test.feature:11
    Given 10 coef a, coef b is 0 and c is 0        # features/steps/test.py:7
    When starting function                         # features/steps/test.py:18
    Then we shoud see "[0]"                        # features/steps/test.py:26

  Scenario Outline: multiple roots roots -- @1.3   # features/test.feature:12
    Given 1 coef a, coef b is -2 and c is -8       # features/steps/test.py:7
    When starting function                         # features/steps/test.py:18
    Then we shoud see "[-2, 2]"                    # features/steps/test.py:26

  Scenario Outline: multiple roots roots -- @1.4   # features/test.feature:13
    Given 4 coef a, coef b is 16 and c is 0        # features/steps/test.py:7
    When starting function                         # features/steps/test.py:18
    Then we shoud see "[-2, 0, 2]"                 # features/steps/test.py:26

  Scenario Outline: multiple roots roots -- @1.5   # features/test.feature:14
    Given 1 coef a, coef b is -10 and c is 9       # features/steps/test.py:7
    When starting function                         # features/steps/test.py:18
    Then we shoud see "[-3, 1, 1, 3]"              # features/steps/test.py:26

1 feature passed, 0 failed, 0 skipped
5 scenarios passed, 0 failed, 0 skipped
15 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.004s
```