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)

**Отчет по лабораторной работе № 4 по курсу**

**Базовые компоненты интернет-технологий**

**«Шаблоны проектирования и модульное тестирование в Python»**

|  |  |
| --- | --- |
| Выполнил: | Проверил: |
| студент группы ИУ5-34Б |  |
| Братухин Александр |  |
| Подпись и дата:  13.12.21 | Подпись и дата: |

Москва, 2021

**Постановка задачи**

1. Необходимо для произвольной предметной области реализовать от одного до трех шаблонов проектирования: один порождающий, один структурный и один поведенческий. В качестве справочника шаблонов можно использовать [следующий каталог.](https://refactoring.guru/ru/design-patterns/catalog) Для сдачи лабораторной работы в минимальном варианте достаточно реализовать один паттерн.
2. Вместо реализации паттерна Вы можете написать тесты для своей программы решения биквадратного уравнения. В этом случае, возможно, Вам потребуется доработать программу решения биквадратного уравнения, чтобы она была пригодна для модульного тестирования.
3. В модульных тестах необходимо применить следующие технологии:
   * TDD - фреймворк.
   * BDD - фреймворк.
   * Создание Mock-объектов.

**Текст программы**

Файл **qr.py**

import sys

import math

def is\_real(string):

    try:

        float(string)

        return True

    except (ValueError, TypeError):

        return False

def check(index, prompt, coef\_str):

    while True:

            if not is\_real(coef\_str):

                print(f"{index} коэффициент введен неверно, невозможно "

                      f"преобразовать в вещественное число\n{prompt} ",end = '')

                coef\_str = input()

            else:

                break

    return coef\_str

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 = check(index, prompt, sys.argv[index])

    except:

        # Вводим с клавиатуры

        print(prompt, end = ' ')

        coef\_str = check(index, prompt, input())

    # Переводим строку в действительное число

    coef = float(coef\_str)

    return coef

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

    '''

    Вычисление корней квадратного уравнения

    Args:

        a (float): коэффициент А

        b (float): коэффициент B

        c (float): коэффициент C

    Returns:

        list[float]: Список корней

    '''

    if a == 0 and b == 0 and c == 0:

        result = [1, 2, 3, 4, 5]

    elif a > 0 and b > 0 and c > 0 or a < 0 and b < 0 and c < 0:

        result = []

    elif a == 0 or b == 0:

        if c > 0:

            result = []

        elif c == 0:

            result = [0,]

        else:

            if a == 0:

                result = [-math.sqrt(-c/b), math.sqrt(-c/b)]

            else:

                result = [-math.sqrt(math.sqrt(-c/a)), math.sqrt(math.sqrt(-c/a))]

    elif c == 0:

        if a > 0 and b > 0:

            result = [0,]

        else:

            result = [-math.sqrt(abs(b/a)), 0, math.sqrt(abs(b/a))]

    else:

        D = b \* b - 4 \* a \* c

        if D < 0:

            result = []

        elif D == 0:

            result = [-math.sqrt(-b / 2 / a), math.sqrt(-b / 2 / a)]

        else:

            D = math.sqrt(D)

            s1 = (-b + D) / 2 / a

            s2 = (-b - D) / 2 / a

            if s1 < 0 and s2 < 0:

                result = []

            elif (s1 > 0 and s2 < 0) or (s1 < 0 and s2 > 0):

                if s1 > 0:

                    result = [-math.sqrt(s1), math.sqrt(s1)]

                else:

                    result = [-math.sqrt(s2), math.sqrt(s2)]

            else:

                result = [-math.sqrt(s1), -math.sqrt(s2), math.sqrt(s2), math.sqrt(s1)]

    return result

def main():

    '''

    Основная функция

    '''

    a = get\_coef(1, 'Введите коэффициент А:')

    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(f"Один корень: {roots[0]}")

    elif len\_roots == 2:

        print(f"Два корня: {roots[0]} и {roots[1]}")

    elif len\_roots == 3:

        print(f"Три корня: {roots[0]}, {roots[1]} и {roots[2]}")

    elif len\_roots == 4:

        print(f"Четыре корня: {roots[0]}, {roots[1]}, {roots[2]} и {roots[3]}")

    else:

        print("Бесконечное множество корней")

# Если сценарий запущен из командной строки

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Файл **testTDD.py**

import unittest

from qr import get\_roots

class TestQr(unittest.TestCase):

    def test1(self):

        self.assertEqual(get\_roots(6, -2, -4), [-1, 1])

    def test2(self):

        self.assertEqual(get\_roots(-7, 1, 6), [-1, 1])

    def test3(self):

        self.assertEqual(get\_roots(1, 5, 6), list())

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

    unittest.main()

Файл **testBDD.py**

from behave import given, when, then

from qr import get\_roots

@given("I have the coefficients {number1}, {number2} and {number3}")

def have\_numbers(context, number1, number2, number3):

    context.number1 = number1

    context.number2 = number2

    context.number3 = number3

@when("I calculate them")

def calculate\_roots(context):

    context.result = set(get\_roots(float(context.number1), float(context.number2), float(context.number3)))

@then("I expect the result to be {result}")

def expect\_result(context, result):

    assert context.result == set(map(float, result.split(',')))

Файл **testBDD.feature**

Feature: Test my equation

    Scenario: Test my equation1

        Given I have the coefficients 4, -5 and 1

        When I calculate them

        Then I expect the result to be -1, -0.5, 0.5, 1

    Scenario: Test my equation2

        Given I have the coefficients 100, 0 and 0

        When I calculate them

        Then I expect the result to be 0

    Scenario: Test my equation3

        Given I have the coefficients 1, -2 and -8

        When I calculate them

        Then I expect the result to be -2, 2

**Пример выполнения программы**

