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

**Факультет Радиотехнический**
Кафедра “Системы обработки информации и управления”

Курс «Парадигмы и конструкции языков программирования»

# Отчет по лабораторной работе №5
# “ Модульное тестирование в Python.”

Выполнил:
студент группы РТ5-31Б:
Каландаров Алим
Шамильевич
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич
Подпись и дата:

Москва, 2025 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.
Я выбрал функцию поиска корней биквадратного уравнения из Лабораторной работы №1.
Разработайте модульные тесты. В модульных тестах необходимо применить следующие технологии:

- TDD - фреймворк (не менее 3 тестов).
- BDD - фреймворк (не менее 3 тестов).

## Текст программы

*Модульный тест по принципу TDD*
*Файл:* Lab5_PIKYAP_TDD.py

```
from pytest_bdd import scenario, given, when, then
import pytest
from lab1PIKYAP import get_biquadratic_roots

# TDD тест
def test_two_roots():
    assert get_biquadratic_roots(1, 0, -1) == [-1.0, 1.0]

def test_one_root_zero():
    assert get_biquadratic_roots(1, 0, 0) == [0.0]

def test_no_real_roots():
    assert get_biquadratic_roots(1, 0, 1) == []

def test_a_equals_zero():
    assert get_biquadratic_roots(0, 1, 1) == []
```

***Модульный тест по принципу BDD***

***Сценарий (словесное описание)***

Feature: Вычисление корней биквадратного уравнения

Для проверки функции get_biquadratic_roots

Scenario: Случай с четырьмя различными действительными корнями
Given коэффициенты биквадратного уравнения a=1, b=-5, c=4
When функция вычисляет корни
Then результат должен содержать корни [1.0, 2.0, -2.0, -1.0]

Scenario: Случай с двумя различными действительными корнями (t1 > 0, t2 < 0)
Given коэффициенты биквадратного уравнения a=1, b=3, c=-4
When функция вычисляет корни
Then результат должен содержать корни [-1.0, 1.0]

Scenario: Случай с одним действительным корнем (t=0)
Given коэффициенты биквадратного уравнения a=1, b=0, c=0
When функция вычисляет корни
Then результат должен содержать корни [0.0]

Scenario: Случай с нулевым коэффициентом 'a'
Given коэффициенты биквадратного уравнения a=0, b=1, c=1
When функция вычисляет корни
Then функция должна вывести сообщение об ошибке "В биквадратном уравнении a не может равняться 0"
And результат должен содержать пустой список []

## Код

```
import pytest
from pytest_bdd import scenarios, given, when, then, parsers
import math

# --- Сама тестируемая функция ---
def get_biquadratic_roots(a, b, c):
    result = []
    if a == 0:
        print("В биквадратном уравнении a не может равняться 0")
        return result
    D = b*b - 4*a*c
    if D < 0.0:
        return result
    elif D == 0.0:
        t = -b / (2.0*a)
        if t > 0:
            root = math.sqrt(t)
            result.append(root)
            result.append(-root)
        elif t == 0:
            result.append(0.0)
    else:
        sqrt_D = math.sqrt(D)
        t1 = (-b + sqrt_D) / (2.0*a)
        t2 = (-b - sqrt_D) / (2.0*a)
        if t1 > 0:
            root = math.sqrt(t1)
            result.append(root)
            result.append(-root)
        elif t1 == 0:
            result.append(0.0)
        if t2 > 0:
            root = math.sqrt(t2)
            result.append(root)
            result.append(-root)
        elif t2 == 0:
            result.append(0.0)
    result = sorted(set(result))
    return result
# --------------------------------

# Загружаем все сценарии из файла find_roots.feature
scenarios('features/find_roots.feature')

@pytest.fixture
```

```
def context():
    return {} # Возвращаем пустой словарь

@given(parsers.parse('коэффициенты биквадратного уравнения a={a:g}, b={b:g},
c={c:g}'))
def step_given_coeffs(context, a, b, c):
    context['a'] = a
    context['b'] = b
    context['c'] = c

@when('функция вычисляет корни')
def step_when_calculates(context):
    # Передаем значения из словаря в функцию
    context['result'] = get_biquadratic_roots(context['a'], context['b'],
context['c'])

@then(parsers.parse('результат должен содержать корни {expected_roots}'))
def step_then_result_contains(context, expected_roots):
    expected_list = [float(x) for x in expected_roots.strip('[]').split(', ')]

    actual_result_sorted = sorted(context['result'])
    expected_list_sorted = sorted(expected_list)

    assert actual_result_sorted == expected_list_sorted, \
        f"Ожидались корни {expected_list_sorted} (отсортированные), получены
{actual_result_sorted} (отсортированные)"



@then(parsers.parse('функция должна вывести сообщение об ошибке "{message}"'))
def step_then_error_message(context, message):
    pass

@then(parsers.parse('результат должен содержать пустой список {empty_list}'))
def step_then_empty_result(context, empty_list):
    assert context['result'] == [], f"Ожидался пустой список, получен
{context['result']}"
```

# Анализ результатов

1. TDD тест

```
C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP>pytest Lab5PIKYAP_TDD.py
============================================== test session starts ===
platform win32 -- Python 3.12.6, pytest-9.0.2, pluggy-1.6.0
rootdir: C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP
plugins: bdd-8.1.0
collected 4 items

Lab5PIKYAP_TDD.py ....

============================================== 4 passed in 0.03s ====

C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP>
```

2.BDD

```
C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP\Lab5>pytest test_roots.py
============================================== test session starts =
platform win32 -- Python 3.12.6, pytest-9.0.2, pluggy-1.6.0
rootdir: C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP\Lab5
plugins: bdd-8.1.0
collected 4 items

test_roots.py ....

============================================== 4 passed in 0.06s ==

C:\Users\Mashiro\Projects\Labs\3_sem\PIKYAP\Lab5>
```