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

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Основы информатики»

# Отчет по лабораторной работе №5

# «Модульное тестирование в Python»

Выполнил:
студент группы ИУ5-35Б
Кулешова Ирина
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Нардид А. Н.
Подпись и дата:

Москва, 2022 г.

## 1. Описание 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:

    a. TDD - фреймворк (не менее 3 тестов).
    b. BDD - фреймворк (не менее 3 тестов).
    c. Создание Mock-объектов (необязательное дополнительное задание).

## 2. Текст программы:

**main.py**

```
import sys
import math


def get_coef(index, prompt):
    try:
        coef_str = sys.argv[index]
    except:
        print(prompt)
        coef_str = input()
    # Переводим строку в действительное число
    '''
    Проверяем можно ли преобразовать строку в число и если
    нельзя, то вводим коэффицент вновь.
    '''
    try:
        coef = float(coef_str)
        # print(string_int)
    except ValueError:
        # Handle the exception
        print('Введено некорректное число.')
        coef = get_coef(index, 'Введите коэффициент снова:')
    return coef
```

```
def get_D(a, b, c):
    return b * b - 4 * a * c

def get_D2(a, b, c):
    return (- b + c) / (2 * a)

def get_roots(a, b, c):
    result = []  # Список корней
    '''
    Рассмотрим случаи, когда один из коэффициентов
    b или c равен 0 отдельно, так как их можно
    вычислить проще.
    '''
    if c == 0:
        result.append(0)
        Dc = - b / a
        if Dc > 0:
            root1 = math.sqrt(Dc)
            root2 = - math.sqrt(Dc)
            result.append(root1)
            result.append(root2)
        return result

    elif b == 0:
        Db = - c / a
        if Db > 0:
            root1 = math.sqrt(math.sqrt(Db))
            root2 = - math.sqrt(math.sqrt(Db))
            result.append(root1)
            result.append(root2)
        if Db == 0:
            result.append(0)
        return result

    else:
        D1 = get_D(a, b, c)
        if D1 < 0:
            return result
        elif D1 == 0:
            D2 = get_D2(a, b, 0)
            if D2 < 0:
                return result
            # Если D2 = 0, то b = 0, а такой случай мы разобрали
            else:
                root1 = - math.sqrt(D2)
                root2 = math.sqrt(D2)
```

```
            result.append(root1)
            result.append(root2)
        return result
    else:
        D3 = (a, b, -math.sqrt(D1))
        if D3 == 0:
            result.append(0)
        if D3 > 0:
            root1 = - math.sqrt(D3)
            root2 = math.sqrt(D3)
            result.append(root1)
            result.append(root2)
        D4 = (a, b, math.sqrt(D1))
        if D4 == 0:
            result.append(0)
        if D4 > 0:
            root3 = - math.sqrt(D4)
            root4 = math.sqrt(D4)
            result.append(root3)
            result.append(root4)
        return result


def main():
    a = get_coef(1, 'Введите коэффициент A:')
    while a == 0:
        a = get_coef(1, 'Коэффицент A не может быть равен 0. Введите
коэффициент A снова:')
    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]))
```

**tdd-tessts.py**

```
import unittest
import math
from main import get_D, get_D2, get_roots

class TestCalculator(unittest.TestCase):
    def test_get_D(self):
        self.assertEqual(get_D(1, 1, 1), -3)
        self.assertEqual(get_D(0.25, 0, -1), 1)
        self.assertEqual(get_D(0, 10, -1000), 100)

    def test_get_D2(self):
        self.assertEqual(get_D2(1, 3, 5), 1)
        self.assertEqual(get_D2(2, 0, 2), 0.5)
        self.assertEqual(get_D2(13, 0, 0), 0)

    def test_get_roots(self):
        self.assertEqual(get_roots(1, -2, 1), [-1, 1])
        self.assertEqual(get_roots(1, 0, 0), [0])
        self.assertEqual(get_roots(1, -4, 0), [0, 2, -2])
        self.assertEqual(get_roots(1, 1, 1), [])

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

**calculation1.feature**

```
Feature: Test calculations

  Scenario Outline: Count discriminant
    Given I have the numbers <num1>, <num2> and <num3>
    When I count discriminant
    Then I expect the result to be <result>

    Examples:
      | num1 | num2 | num3 | result |
      |  1   |  1  |  1   |  -3   |
      |  0.25   |  0  |  -1  |  1  |
      |  0   |  10  |  -1000  |  100  |
```

**bdd-tests.py**

```
import builtins
```

```
import pathlib
from pathlib import Path
from pytest_bdd import scenario, scenarios, given, when, then, parsers
from main import get_D, get_D2, get_roots

featureFileDir = 'my_features'
featureFile = 'calculation1.feature'
BASE_DIR = Path(__file__).resolve().parent
FEATURE_FILE = BASE_DIR.joinpath(featureFileDir).joinpath(featureFile)

#scenarios("calculation2.feature")

@given(parsers.parse("I have the numbers {num1:d}, {num2:d} and {num3:d}"))
def constants(num1, num2, num3):
    const = [num1, num2, num3]
    return const

@when("I count discriminant")
def counting(const):
    res = const[1] * const[1] - 4 * const[0] * const[2]
    return res

@then(parsers.parse("I expect the result to be {result:d}"))
def comparison(result):
    assert get_D(1, 1, 1) == result

if __name__ == "__main__":
    scenarios("calculation2.feature")
```

## 3. Экранные формы с примерами выполнения программы:

```
C:\Users\Ирина\AppData\Local\Programs\Python\Python311\python.exe "D:/Program Files/Jet
Testing started at 7:36 ...
Launching pytest with arguments C:/Users/Ирина/PycharmProjects/lab5_1/tdd-tests.py --no

============================= test session starts =============================
collecting ... collected 3 items

tdd-tests.py::TestCalculator::test_get_D PASSED                          [ 33%]
tdd-tests.py::TestCalculator::test_get_D2 PASSED                         [ 66%]
tdd-tests.py::TestCalculator::test_get_roots PASSED                      [100%]

============================== 3 passed in 0.03s ==============================

Process finished with exit code 0
```