Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №4
«Модульное тестирование в Python»

Выполнил:
студент группы ИУ5-32Б
Канаева Д.Ч.
Подпись и дата:

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

Цель лабораторной работы: изучение возможностей модульного тестирования в языке Python.

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.

EquationSolver.py

```
import math

class EquationSolver:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c

    def solve(self):
        dis = self.b**2 - 4*self.a*self.c
        if dis >= 0.0:
            x1_1 = (-self.b + math.sqrt(dis))/(2*self.a)
            flag1 = 0
            flag2 = 0

            if x1_1 >= 0:
                x1 = math.sqrt(x1_1)
                x2 = x1 * (-1)
                flag1 = 1

            x3_3 = (-self.b - math.sqrt(dis))/(2*self.a)
            if x3_3 >= 0:
                x3 = math.sqrt(x3_3)
                x4 = x3 * (-1)
                flag2 = 1

            if flag1 and flag2:
                return x1, x2, x3, x4
            elif flag1:
                return x1, x2
            elif flag2:
                return x3, x4

        return None
```

tests/test_equation_solver.py

```
import sys
import os

sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
import EquationSolver

def test_zero_discr():
    solver = EquationSolver.EquationSolver(1, -2, 1)
    result = solver.solve()
    assert result == (1.0, -1.0, 1.0, -1.0)

def test_negative_discr():
    solver = EquationSolver.EquationSolver(1, 1, 1)
    result = solver.solve()
    assert result == None
```

features/equation_solver.feature

```
Feature: Equation Solver

Scenario: Solving equation with zero discriminant
  Given I have an equation solver with coefficients 1, -2, 1
  When I solve the equation
  Then I expect the result to be 1.0 and -1.0 and 1.0 and -1.0

Scenario: Solving equation with negative discriminant
  Given I have an equation solver with coefficients 1, 1, 1
  When I solve the equation
  Then I expect the result to be None
```

features/steps/equation_solver_steps.py

```
from behave import given, when, then
import sys
import os
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
from EquationSolver import EquationSolver


@given('I have an equation solver with coefficients {a:d}, {b:d}, {c:d}')
def step_given_equation_solver(context, a, b, c):
    context.solver = EquationSolver(a, b, c)


@when('I solve the equation')
def step_when_solve_equation(context):
    context.result = context.solver.solve()


@then('I expect the result to be {expected_result}')
def step_then_check_result(context, expected_result):
    if expected_result == 'None':
        expected_result = None
    else:
        expected_result = tuple(float(x) for x in
expected_result.split('and'))

    assert context.result == expected_result
```

Результаты выполнения программ:

```
C:\Users\123\Documents\PCPL_3sem\Lab4>pytest
============================================= test session starts =============================================
platform win32 -- Python 3.10.0, pytest-7.4.3, pluggy-1.3.0
rootdir: C:\Users\123\Documents\PCPL_3sem\Lab4
collected 2 items

tests\test_equation_solver.py ..                                                                      [100%]

============================================== 2 passed in 0.01s ==============================================
```

```
C:\Users\123\Documents\PCPL_3sem\Lab4>behave
Feature: Equation Solver # features/equation_solver.feature:1

  Scenario: Solving equation with zero discriminant              # features/equation_solver.feature:3
    Given I have an equation solver with coefficients 1, -2, 1   # features/steps/equation_solver_steps.py:8
    When I solve the equation                                    # features/steps/equation_solver_steps.py:12
    Then I expect the result to be 1.0 and -1.0 and 1.0 and -1.0 # features/steps/equation_solver_steps.py:16

  Scenario: Solving equation with negative discriminant       # features/equation_solver.feature:8
    Given I have an equation solver with coefficients 1, 1, 1 # features/steps/equation_solver_steps.py:8
    When I solve the equation                                 # features/steps/equation_solver_steps.py:12
    Then I expect the result to be None                       # features/steps/equation_solver_steps.py:16

1 feature passed, 0 failed, 0 skipped
2 scenarios passed, 0 failed, 0 skipped
6 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.001s
```