Московский государственный технический университет имени Н.Э.Баумана

Факультет РТ Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

Отчет по лабораторной работе № 4 по курсу Базовые компоненты интернет-технологий

Исполнитель

Студент группы РТ5-31Б _____ Дворкович Ю.А.

“___” _____ 2021 г.

Проверил

Доцент кафедры ИУ5 _____ Гапанюк Ю.Е.

“___” _____ 2021 г.

Москва
2021

Содержание:

1.Описание задания

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. 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.

2.Текст программы

Mock.py

```
import unittest

from unittest.mock import Mock

from roots import get_roots

class TestQr(unittest.TestCase):

        def test_get_roots(self):

                mockRoot = Mock(return_value = 412)

                get_roots(mockRoot(), 2, 1)

if __name__ == "__main__":

        unittest.main()
```

TDD.py

```
import unittest

from roots import get_roots

class TestGetRoots(unittest.TestCase):

   def testGetRoots(self):

            self.assertEqual(get_roots(1, -10, 21), [7, 3])

            self.assertEqual(get_roots(1, 2, 1), [-1])
```

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

testBDD.py

```
from behave import *
from roots import get_roots
@given('I put roots {roots} into the function')
def step_impl(context, roots: str):
   roots = list(map(float, roots[1:-1].split(',')))
   context.result = get_roots(*roots)
@then('I get roots {result}')
def step_impl(context, result: str):
   result = list(map(float, result[1:-1].split(',')))
   assert sorted(context.result) == sorted(result)
```

BDD.feature

```
Feature: testing the function get_roots
        Scenario: get roots of BQ
                Given I put roots [1, -2, 1] into the function
                Then I get roots [-1.0, 1.0]
        Scenario: get roots of BQ
                Given I put roots [1, -10, 9] into the function
                Then I get roots [3.0, -3.0, 1.0, -1.0]
```

3.Экранные формы с примерами выполнения программы

```
Feature: testing the function get_roots # BDD.feature:1

  Scenario: get roots of BQ                       # BDD.feature:2
    Given I put roots [1, -2, 1] into the function # steps/testBDD.py:5
    Then I get roots [-1.0, 1.0]                  # steps/testBDD.py:11

  Scenario: get roots of BQ                        # BDD.feature:6
    Given I put roots [1, -10, 9] into the function # steps/testBDD.py:5
    Then I get roots [3.0, -3.0, 1.0, -1.0]         # steps/testBDD.py:11

1 feature passed, 0 failed, 0 skipped
2 scenarios passed, 0 failed, 0 skipped
4 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.000s
```