МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий

Кафедра КБ-3 «Разработка программных решений и системного программирования»

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №1

по дисциплине: «Сборка, тестирование и верификация программного продукта»

на тему: «doctest vs unittest»

Выполнил: студент группы БСБО-10-21

Филёв С. А.

Проверил: преподаватель кафедры КБ-3

Ивакин С. Н.

Москва, 2024 г

Ход работы

Была выбрана математическая функция нахождения факториала числа (см. листинг 1).

Листинг 1. Функция нахождения алгоритма с примерами

```
def factorial(n):
    """
    Вычисляет факториал числа n.

    Функция поддерживает только неотрицательные целые числа.

    Примеры:
    >>> factorial(5)
    120
    >>> factorial(0)
    1
    >>> factorial(1)
    1
    >>> factorial(3)
    6
    >>> factorial(-1)
    Traceback (most recent call last):
        ...
    ValueError: Factorial is only defined for non-negative
integers
    """
    if n < 0:
        raise ValueError("Factorial is only defined for non-
negative integers")
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result
```

Далее в листингах 2, 3 и 4 были прописаны тесты данной функции, а именно использовав конструкции assert, doctest и unittest, а также результаты проведения тестов на рисунках 1, 2, 3 соответственно.

Листинг 2. Тест функции с помощью assert

```
if __name__ == '__main__':
    # Основные тесты для функции factorial
    assert factorial(5) == 120, "Ошибка: factorial(5) должен
вернуть 120"
    assert factorial(0) == 1, "Ошибка: factorial(0) должен
вернуть 1"
    assert factorial(1) == 1, "Ошибка: factorial(1) должен
вернуть 1"
    assert factorial(3) == 6, "Ошибка: factorial(3) должен
вернуть 6"

    # Проверка на ошибку при вводе отрицательного числа
    try:
        factorial(-1)
    except ValueError as e:
        assert str(e) == "Factorial is only defined for non-
negative integers", \
            "Ошибка: неправильное сообщение об ошибке для
factorial(-1)"
```

Листинг 3. Тест функции с помощью doctest

```
if __name__ == '__main__':
    import doctest
    doctest.testmod()
```

Листинг 4. Тест функции с помощью unittest

```
import unittest
from pr_1_1 import factorial
```

```
class TestFactorial(unittest.TestCase):

    def test_positive_numbers(self):
        self.assertEqual(factorial(5), 120)
        self.assertEqual(factorial(3), 6)
        self.assertEqual(factorial(1), 1)

    def test_zero(self):
        self.assertEqual(factorial(0), 1)

    def test_negative_numbers(self):
        with self.assertRaises(ValueError):
            factorial(-1)


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

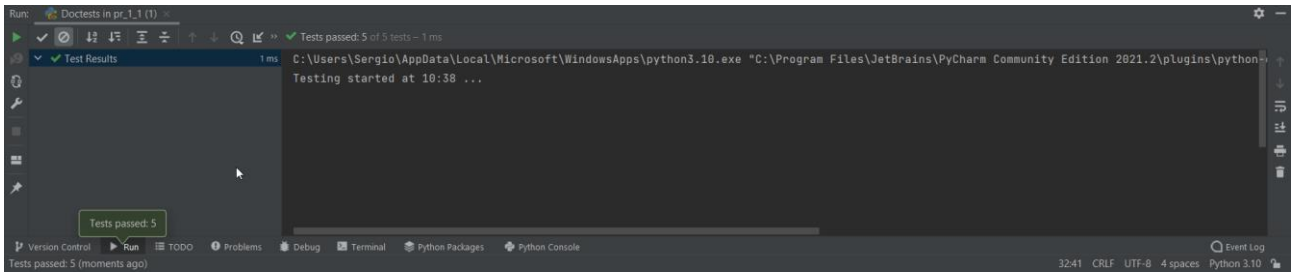

Рисунок 1 – Тестирование с помощью assert

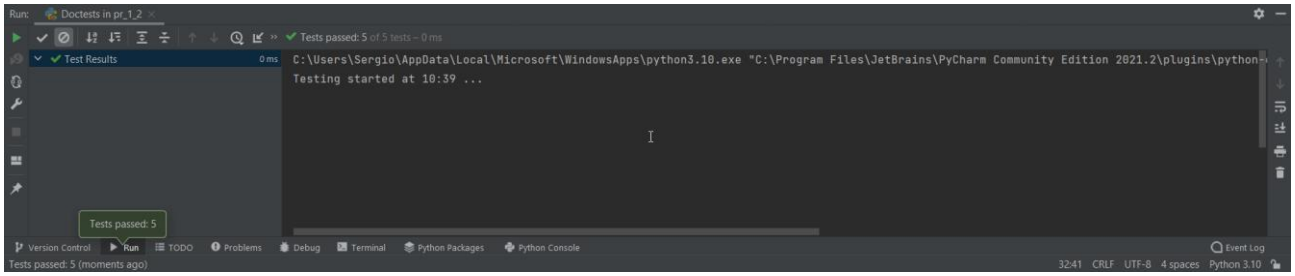

Рисунок 2 – Тестирование с помощью doctest

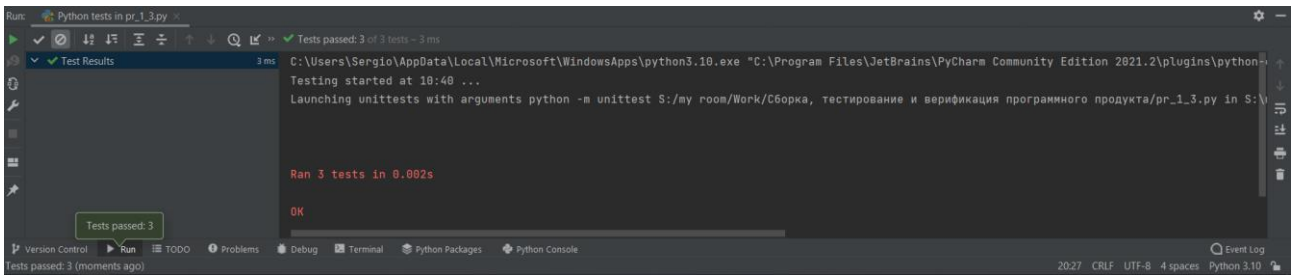

Рисунок 3 – Тестирование с помощью unittest

1. Основные случаи: n = 0, n = 1, небольшие положительные значения.
2. Крайние случаи: отрицательные значения, при которых функция должна выдавать ошибку ValueError.
3. Простая проверка: n = 1 и n = 0, где результат предсказуем — 1.