Федеральное государственное автономное

образовательное учреждение1

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
|  |
| Институт космических и информационных технологий |

институт

|  |
| --- |
| Информатика |

кафедра

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №5**

|  |
| --- |
| Тестирование кода |

тема

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | П В Пересунько |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-17/1б, 032156946 |  |  |  | Д А Князьков |
|  | номер группы, зачетной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2021

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc90811403)

[1 Цель 3](#_Toc90811404)

[2 Задачи 3](#_Toc90811405)

[3 Описание варианта задания 4](#_Toc90811406)

[4 Ход работы 4](#_Toc90811407)

[5 Вывод 6](#_Toc90811408)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 7](#_Toc90811409)

1 Цель

Изучить принципы работы модульного тестирования в Python. Научиться писать модульные тесты (в том числе параметризированные тесты) с использованием пакетов PyTest, создавать фикстуры тестов, проверять полноту покрытия при помощи инструмента coverage.

1. Задачи
2. Оформить код из практической №4 в виде отдельного пакета, а также предусмотреть рядом с исходным кодом директорию «tests/» или просто файл с тестами;
3. Создать отдельную ветку, на которой будет производиться написание тестов для вашего кода;
4. Написать набор модульных тестов (не менее пяти тестов для покрытия всех частных случаев входных данных) для функций или методов класса из практической №4 в соответствии со своим вариантом;
5. Продемонстрировать понимание и владение навыками работы с фикстурами и параметризированными тестами;
6. Проверить полноту покрытия тестами исходного кода программы при помощи соответствующего инструмента;
7. В случае 100%-го покрытия тестами необходимо приложить скриншот после запуска coverage
8. В случае неполного покрытия тестами, в дополнение к предыдущему, пункту, показать незадействованный блок кода и новые тесты для него;
9. Слить ветку с тестами в главную рабочую ветку;
10. Оформить отчет по практической работе;

3 Описание варианта задания

11-ый вариант.

Необходимо написать две функции: первая будет определять, является ли заданная сеть полносвязной; вторая будет определять, к какому типу неполносвязных сетей относится заданная сеть.

На вход каждой функции подается сначала два целых положительных числа V и R – число вершин и рёбер в графе соответственно (𝑉 > 4, 𝑅 > 3), а затем массив P из кортежей целых чисел (i, j), где пара i, j – наличие связи между вершинами под номерами i и j (𝑉 > 𝑖 > 0, 𝑉 > 𝑗 > 0).

Функции обязательно должны соответствовать определённым сигнатурам, представленным на листингах 1-2

Листинг 1 – Сигнатура первой функции

def is\_full\_connected (v: int, r: int, links: List[Tuple[int, int]]) -> boolean:

Листинг 2 – Сигнатура второй функции

def connection\_type (v: int, r: int, links: List[Tuple[int, int]]) -> int:

4 Ход работы

В ходе выполнения данной практической работы было написано несколько функций для тестирования программы из практической работы №4

Листинг 3 – Тестовые функции

@pytest.mark.parametrize("v, r, links", [(4, 12, [(1, 2), (3, 4)]), (3, 6, [(1, 3)]), (5, 20, [(1, 2), (1, 3)])])

def test\_is\_full\_connected(v, r, links):

assert is\_full\_connected(v, r, links) is True

@pytest.mark.parametrize("v, r, links",

[(6, 5, [(1, 2), (2, 3), (3, 4), (4, 5), (5, 6)]), (4, 3, [(1, 2), (2, 3), (3, 4)])])

def test\_connection\_type\_1(v, r, links):

assert connection\_type(v, r, links) == 1

@pytest.mark.parametrize("v, r, links",

[(5, 5, [(1, 2), (2, 3), (3, 4), (4, 5), (1, 5)]), (3, 3, [(1, 2), (2, 3), (3, 1)])])

def test\_connection\_type\_2(v, r, links):

assert connection\_type(v, r, links) == 2

@pytest.mark.parametrize("v, r, links", [(5, 4, [(1, 2), (1, 3), (1, 4), (1, 5)]), (4, 3, [(1, 2), (1, 3), (1, 4)])])

def test\_connection\_type\_3(v, r, links):

assert connection\_type(v, r, links) == 3

@pytest.mark.parametrize("v, r, links",

[(4, 5, [(1, 2), (1, 3), (1, 4), (2, 1), (2, 3)]), (4, 4, [(1, 2), (1, 3), (2, 3), (2, 4)]),

(4,6,[(1,2),(1,3),(1,4),(3,4),(2,3)])])

def connection\_type\_4(v, r, links):

assert connection\_type(v, r, links) == -1

После прохождения всех тестов была выполнена проверка на покрытие тестами основной программы

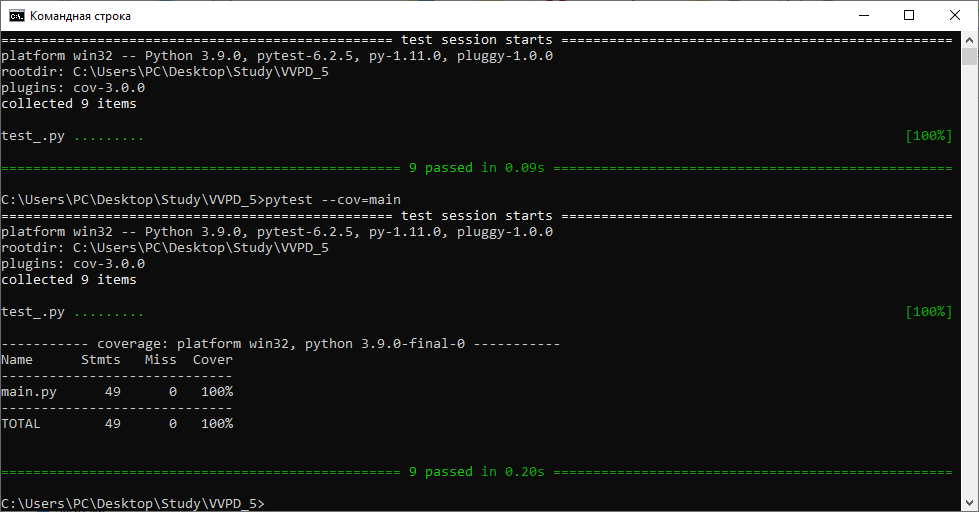


Рисунок 1 – Покрытие тестами программы из практической работы №4

В результате было получено 100% покрытие тестами основной программы. После этого ветка tests, в которой создавались тесты, была слита с основной рабочей веткой.

5 Вывод

В ходе выполнения данной практической работы были изучены принципы работы модульного тестирования в Python и созданы тесты для программы из практической работы №4 с использованием модуля pytest

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Официальная документация Git: <https://git-scm.com/book/ru/v2>

2) Официальная документация pytest: docs.pytest.org