## Министерство Образования и Исследований

## Республики Молдова

## Технический Университет Молдовы

## Факультет Вычислительной Техники, Информатики и Микроэлектроники

## Отчет

## по лабораторной работе Nr.6. по:

## Декларативному Программированию

## 

## 

## Выполнила ст.гр. TI-219: Галупа Яна

## Проверил: Лях Аркадий

## 

## Кишинев – 2023

**Ссылка на git**

1. Создайте тестовый модуль (unittest) для вашего модуля реализованного в 5 лабораторной [название модуля].
2. Напишите несколько тестовых кейсов для функций, которые требуется протестировать.
3. Запустите тестовый модуль для проверки правильности работы функций в модуле [название вашего модуля].
4. Удостоверьтесь, что все тесты проходят успешно.

import unittest

import os

from file\_utils import get\_file\_size, copy\_file

class TestFileUtils(unittest.TestCase):

    def setUp(self):

        # Creating temporary files for testing

        self.source\_path = 'source.txt'

        self.destination\_path = 'destination.txt'

        with open(self.source\_path, 'w') as file:

            file.write("This is a test file.")

    def tearDown(self):

        # Removing temporary files after testing

        if os.path.exists(self.source\_path):

            os.remove(self.source\_path)

        if os.path.exists(self.destination\_path):

            os.remove(self.destination\_path)

    def test\_get\_file\_size(self):

        # Checking file size retrieval

        size = get\_file\_size(self.source\_path)

        self.assertEqual(size, 20)

    def test\_get\_file\_size\_nonexistent\_file(self):

        # Checking for non-existent file

        size = get\_file\_size('nonexistent\_file.txt')

        self.assertEqual(size, -1)

    def test\_copy\_file(self):

        # Checking file copying

        result = copy\_file(self.source\_path, self.destination\_path)

        self.assertEqual(result, "File successfully copied.")

        # Checking if the copied file exists

        self.assertTrue(os.path.exists(self.destination\_path))

        # Checking the content of the copied file

        with open(self.destination\_path, 'r') as file:

            content = file.read()

        self.assertEqual(content, "This is a test file.")

    def test\_copy\_file\_nonexistent\_source(self):

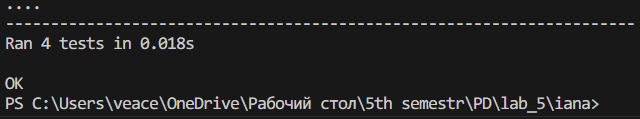
        # Checking copying of a non-existent file

        result = copy\_file('nonexistent\_source.txt', self.destination\_path)

        self.assertEqual(result, "File not found")

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()



Эти тесты проверяют функциональность модуля file\_utils.py:

* test\_get\_file\_size: Проверяет получение размера существующего файла.
* test\_get\_file\_size\_nonexistent\_file: Проверяет получение размера для файла, который не существует.
* test\_copy\_file: Проверяет копирование содержимого из одного файла в другой и сравнивает содержимое скопированного файла.
* test\_copy\_file\_nonexistent\_source: Проверяет копирование файла, который не существует.

Эти тесты помогут обеспечить корректную работу функций модуля file\_utils.py в различных сценариях использования.

**Вывод:**

Результаты лабораторной работы, представленной в файле lab\_6.py, демонстрируют успешное тестирование функциональности модуля file\_utils.py, который включает в себя две основные функции для работы с файлами: get\_file\_size() и copy\_file().

test\_get\_file\_size: Этот тест проверяет корректность получения размера существующего файла. Размер файла совпадает с ожидаемым значением.

test\_get\_file\_size\_nonexistent\_file: В данном тесте проверяется поведение функции при попытке получить размер несуществующего файла. Ожидается, что будет возвращено значение -1.

test\_copy\_file: Этот тест проверяет успешность операции копирования содержимого из одного файла в другой. После копирования содержимое целевого файла совпадает с содержимым исходного файла.

test\_copy\_file\_nonexistent\_source: Данный тест проверяет обработку функцией случая, когда исходный файл для копирования отсутствует. Ожидается, что будет возвращено сообщение об ошибке о том, что файл не найден.

Эти тесты позволяют убедиться в корректной работе функций модуля file\_utils.py, в том числе при обработке краевых случаев, таких как отсутствие файла или успешное копирование содержимого между файлами. Успешное прохождение тестов подтверждает функциональность и надежность модуля file\_utils.py.