

## **Звіт про виконання лабораторної роботи №2**

Тема:Робота з файлами. Юніт тести.

Мета: Використовуючи теоретичне підґрунтя про роботу з файлами та тестування коду у мові Python розширити програму телефонного довідника студентів додавши функціонал, що буде вказано в завданні до лабораторної роботи.

### **Завдання**

Розширити функціональність телефонного довідника студентів групи, що був розроблений у Лабораторній роботі №1 наступними можливостями:

1)завантаження початкових дані для довідника з csv файлу. Приклад csv файлу (lab2.csv), та оригінальний файл формату xlsx розміщені поряд з вказівками. Ім'я файлу для завантаження даних повинно передаватись в якості параметрів командного рядку.

2)зберігання довідника студентів у форматі CSV перед виходом із програми.

3)покриття основного функціоналу програми Юніт тестами.

### **Хід роботи**

1) Спочатку я створив файл lab2.csv, а потім в програму додав пустий масив, і за допомогою argv зробив перевірку параметрів командного рядка, якщо там вказаний csv файл, то його запускати, якщо ні, то автоматично запускати lab\_02.csv і одразу після запуску переписав дані з таблицки в список, помістив всю логіку завантаження csv в окрему функцію(Рис.1).

```
def loadCSV():
    global filename
    if len(argv) > 1:
        filename = argv[1]

    try:
        with open(filename, "r") as file:
            reader = csv.DictReader(file, skipinitialspace=True)
            for row in reader:
                list.append({
                    "name": row["StudentName"],
                    "surname": row["StudentSurname"],
                    "phone": row["StudentPhone"],
                    "email": row["StudentEmail"]
                })
    except FileNotFoundError:
        print("Файл не знайдено, починаємо з порожнім списком.")
    except KeyError as e:
        print(f"Помилка структури CSV: {e}")
```

Рис.1 Функція завантаження даних з csv

2) Для збереження довідника студентів я зробив окрему функцію saveList в якій переписую дані в таблиці, створюю заново заголовки і потім переписую всі дані з списку в таблицю і приміняю цю функцію при виході з програми(Рис.2).

```
def saveList():
    with open(filename, "w") as file:
        fieldnames = ["StudentName", "StudentSurname", "StudentPhone", "StudentEmail"]
        writer = csv.DictWriter(file, fieldnames=fieldnames)
        writer.writeheader()

        for student in list:
            writer.writerow({
                "StudentName": student['name'],
                "StudentSurname": student['surname'],
                "StudentPhone": student['phone'],
                "StudentEmail": student['email'],
            })
    print(f"Дані збережено в {filename}")
```

Рис.2 Функція збереження даних у csv

3)Для того щоб зробити юніт тести я створив окремий файл test\_lab\_02 в який імпортував програму з файлу lab\_02, встановив пакет pytest, для

очищення списку використав декоратор `fixture` щоб уникнути помилок при тестах, далі я створив для кожної функції окремий тест, та використав `monkeypatch` щоб програма не чекала на введення тексту і сама вводила текст в інпут і тест на збереження роблю в окремий файл `csv`.

### Код тестів

```
import pytest
import lab_02

@pytest.fixture(autouse=True)
def clean_list():
    lab_02.list = []
    yield
    lab_02.list = []

def test_add_new_element(monkeypatch):
    inputs = iter(["Ivan", "Petrenko", "12345", "ivan@mail.com"])

    monkeypatch.setattr('builtins.input', lambda _: next(inputs))

    lab_02.addNewElement()

    assert len(lab_02.list) == 1
    assert lab_02.list[0]['name'] == "Ivan"
    assert lab_02.list[0]['surname'] == "Petrenko"

def test_add_element_sorting(monkeypatch):
    inputs1 = iter(["Zara", "A", "1", "1"])
    monkeypatch.setattr('builtins.input', lambda _: next(inputs1))
    lab_02.addNewElement()

    inputs2 = iter(["Andriy", "B", "2", "2"])
    monkeypatch.setattr('builtins.input', lambda _: next(inputs2))
    lab_02.addNewElement()

    assert lab_02.list[0]['name'] == "Andriy"
    assert lab_02.list[1]['name'] == "Zara"

def test_delete_element(monkeypatch):
```

```
lab_02.list.append({"name": "Oleg", "surname": "D", "phone": "1",
"email": "e"})

monkeypatch.setattr('builtins.input', lambda _: "Oleg")
lab_02.deleteElement()

assert len(lab_02.list) == 0

def test_update_element(monkeypatch):
    lab_02.list.append({"name": "Taras", "surname": "Old", "phone":
"000", "email": "student@mail"})

    inputs = iter(["Taras", "", "NewSurname", "", ""])
    monkeypatch.setattr('builtins.input', lambda _: next(inputs))

    lab_02.updateElement()

    assert lab_02.list[0]['surname'] == "NewSurname"
    assert lab_02.list[0]['name'] == "Taras"

def test_save_list():
    lab_02.list = [{"name": "TestUser", "surname": "S", "phone": "1",
"email": "e"}]
    lab_02.filename = "test_output.csv"

    lab_02.saveList()

    with open("test_output.csv", "r") as f:
        content = f.read()
        assert "TestUser" in content
        assert "StudentName" in content
```

**Висновок:** У цій лабораторній роботі я реалізував збереження та завантаження даних довідника у форматі CSV, а також навчився писати юніт-тести за допомогою бібліотеки pytest. Для перевірки функцій вводу/виводу було успішно використано фікстури та імітацію даних (monkeypatch).