Пуховий Сергій КБ-241

# Звіт про виконання лабораторної роботи №3

Тема:ООП.

Мета: Використовуючи теоретичне підґрунтя про ООП у мові Python переробити програму телефонного довідника студентів використовуючи принципи ООП для формування відомостей про студентів.

## Завдання

Переробити функціональність телефонного довідника студентів групи, що був розроблений у Лабораторній роботі №2 використовуючи принципи ООП:

1. розробити клас Студент групи з відповідними атрибутами;
2. розробити клас Список групи, має містити не словники, як виконано в лабораторній роботі №2, а об’єкти класу Студент групи; додавання нового запису, видаленні існуючого чи зміна даних має бути виконана через методи класу Список групи.
3. розробити клас для роботи з файлами для зчитування початкової інформації про список групи та збереження інформації по завершенню програми.
4. список студентів має містити не словники, як виконано в лабораторній роботі №2, а об’єкти класу Студент групи;
5. описання всіх класів мають міститися в окремих файлах, що мають відповідні імена(наприклад Studen, StudentList, Utils)
6. основний функціонал програми має бути покритий Юніт тестами.

## Хід роботи

1) Спочатку я розробив в окремому файлі клас Student в якому ініціалізував всі змінні, такі як ім’я , фамілія, телефон і електронну пошту

```
class Student:
    def __init__(self, name, surname, phone, email):
        self.name = name
        self.surname = surname
        self.phone = phone
        self.email = email

    def __str__(self):
        return f"Name: {self.name} Surname: {self.surname}, Phone:
{self.phone}, Email: {self.email}"
```

2) В клас  StudentList я преніс функції, які створюють видаляють, оновлються та виводять список студентів, спочатку при створенні студента йому задається клас student, а також при оновленні інформації про студента, бо там видаляється старий запис і додається оновлений.

```
from student import Student

class StudentList:
    def __init__(self):
        self.students = []

    def add_student(self, student: Student):
        insert_position = 0
        for item in self.students:
            if student.name > item.name:
                insert_position += 1
            else:
                break
        self.students.insert(insert_position, student)
        print("Student has been added")

    def delete_student(self, name):
        for item in self.students:
            if item.name == name:
                self.students.remove(item)
                print(f"Student {name} has been deleted")
```

```
                return
        print("Element was not found")

    def find_student(self, name):
        for item in self.students:
            if item.name == name:
                return item
        return None

    def update_student(self, old_name, new_student: Student):
        student_to_remove = self.find_student(old_name)
        if student_to_remove:
            self.students.remove(student_to_remove)
            self.add_student(new_student)
            print("Student has been updated")
        else:
            print("Student not found")

    def get_all_students(self):
        return self.students

    def print_all(self):
        for student in self.students:
            print(student)
```

3)Розробив клас FileManager який займається збереженням і виведенням інформації з файлу csv перша функція виводить дані з файлу в термінал, якщо не має такого файлу, то починається програма з пустим списком, який можна заповнити і з'явиться файл, друга функція відповідає за збереження інформації в файлі, проходиться по кожному студенту і записує їх в csv файл.

```
import csv
from student import Student

class FileManager:
    def __init__(self, filename):
        self.filename = filename

    def load_data(self):
        students = []
```

```
        try:
            with open(self.filename, "r", newline='', encoding='utf-8')
as file:
                reader = csv.DictReader(file, skipinitialspace=True)
                for row in reader:
                    student = Student(
                        name=row["StudentName"],
                        surname=row["StudentSurname"],
                        phone=row["StudentPhone"],
                        email=row["StudentEmail"]
                    )
                    students.append(student)
        except FileNotFoundError:
            print("Файл не знайдено, починаємо з порожнім списком.")
        except KeyError as e:
            print(f"Помилка структури CSV: {e}")
        return students

    def save_data(self, student_list):
        with open(self.filename, "w", newline='', encoding='utf-8') as
file:
            fieldnames = ["StudentName", "StudentSurname",
"StudentPhone", "StudentEmail"]
            writer = csv.DictWriter(file, fieldnames=fieldnames)
            writer.writeheader()

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

4) В головному файлі програми я імпортую всі класи і задаю назву csv файлу створюю об’єкти класу FileManager і StudentList, потім завантажую дані з таблиці і виконую цикл самої програми де можна з нею взаємодіяти при створенні предаю об’єкт класу Student до класу StudentList і при оновлені також.

```
import sys
from student import Student
from student_list import StudentList
from utils import FileManager

def main():
    filename = "lab3.csv"
    if len(sys.argv) > 1:
        filename = sys.argv[1]

    file_manager = FileManager(filename)
    stud_list = StudentList()

    loaded_data = file_manager.load_data()
    for s in loaded_data:
        stud_list.add_student(s)

    while True:
        choice = input("Please specify the action [ C create, U update,
D delete, P print, X exit ]: ")
        match choice.lower():
            case "c":
                print("New element will be created:")
                name = input("Please enter student name: ")
                surname = input("Please enter student surname: ")
                phone = input("Please enter student phone: ")
                email = input("Please enter student email: ")
                new_student = Student(name, surname, phone, email)
                stud_list.add_student(new_student)
                stud_list.print_all()

            case "u":
                print("Existing element will be updated")
                name = input("Please enter name to be updated: ")
                found = stud_list.find_student(name)

                if found:
                    print(f"Updating student: {found}")
                    new_name = input(f"Enter new name [{found.name}]:
") or found.name
                    new_surname = input(f"Enter new surname
[{found.surname}]: ") or found.surname
```

```
                    new_phone = input(f"Enter new phone
[{found.phone}]: ") or found.phone
                    new_email = input(f"Enter new email
[{found.email}]: ") or found.email

                    updated_student = Student(new_name, new_surname,
new_phone, new_email)
                    stud_list.update_student(name, updated_student)
                    stud_list.print_all()
                else:
                    print("Student not found")

            case "d":
                print("Element will be deleted")
                name = input("Please enter name to be deleted: ")
                stud_list.delete_student(name)
                stud_list.print_all()

            case "p":
                print("List will be printed")
                stud_list.print_all()

            case "x":
                print("Exit()")
                file_manager.save_data(stud_list.get_all_students())
                break

            case _:
                print("Wrong choice")

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

5)Зробив файл з тестуванням, передав туди всі класи і провів 7 тестів це тести додавання, сортування при додаванні, видалення, видалення не існуючого об’єкта, оновлення, сортування після оновлення та збереження, створення кожного студенту відбувалося за допомогою класу Student, список перевірявся за допомогою класу StudentList і використовую клас

FileManager для тимчасового файлу при тестуванні збереження і читання csv файлу.

```
import pytest
from student import Student
from student_list import StudentList
from utils import FileManager

@pytest.fixture
def student_list():
    return StudentList()

def test_add_student(student_list):
    s1 = Student("Ivan", "Petrenko", "123", "ivan@mail")
    student_list.add_student(s1)

    assert len(student_list.students) == 1
    assert student_list.students[0].name == "Ivan"

def test_add_student_sorting(student_list):
    s1 = Student("Zara", "A", "1", "1")
    s2 = Student("Andriy", "B", "2", "2")

    student_list.add_student(s1)
    student_list.add_student(s2)

    assert student_list.students[0].name == "Andriy"
    assert student_list.students[1].name == "Zara"

def test_delete_student(student_list):
    s1 = Student("Oleg", "D", "1", "e")
    student_list.add_student(s1)

    student_list.delete_student("Oleg")
    assert len(student_list.students) == 0

def test_delete_non_existent(student_list):
    s1 = Student("Oleg", "D", "1", "e")
    student_list.add_student(s1)

    student_list.delete_student("NotOleg")
    assert len(student_list.students) == 1
```

```python
def test_update_student(student_list):
    s1 = Student("Taras", "Old", "000", "mail")
    student_list.add_student(s1)

    s_new = Student("Taras", "NewSurname", "000", "mail")
    student_list.update_student("Taras", s_new)

    assert len(student_list.students) == 1
    assert student_list.students[0].surname == "NewSurname"

def test_update_student_resorting(student_list):
    s1 = Student("Anna", "X", "1", "1")
    s2 = Student("Boris", "Y", "2", "2")
    student_list.add_student(s1)
    student_list.add_student(s2)

    s_updated = Student("Zlatan", "X", "1", "1")
    student_list.update_student("Anna", s_updated)

    assert student_list.students[0].name == "Boris"
    assert student_list.students[1].name == "Zlatan"

def test_file_save_load(tmp_path):
    file_path = tmp_path / "test_data.csv"
    fm = FileManager(str(file_path))
    sl = StudentList()

    s1 = Student("TestUser", "S", "1", "e")
    sl.add_student(s1)

    fm.save_data(sl.get_all_students())

    new_sl = StudentList()
    loaded_students = fm.load_data()
    for s in loaded_students:
        new_sl.add_student(s)

    assert len(new_sl.students) == 1
    assert new_sl.students[0].name == "TestUser"
```

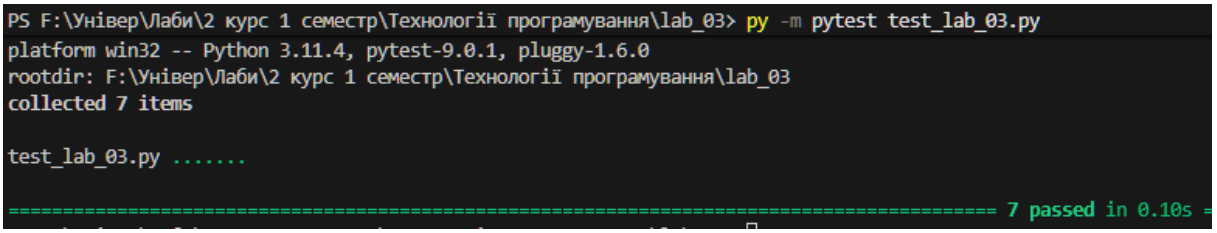


Рис.1 Успішне проходження тесту

**Висновок**: У ході лабораторної роботи я опанував принципи об'єктно-орієнтованого програмування (ООП) у Python. Програму телефонного довідника було успішно рефакторизовано: замість словників використано об'єкти класу Student, логіку управління списком винесено в клас StudentList, а роботу з файлами — у FileManager. Код розподілено по окремих модулях та покрито юніт-тестами, що зробило структуру проекту більш гнучкою, читабельною та надійною.