**Лабораторна робота №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 до лабораторної роботи**

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, видалення 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**

Створив файл main.py у якій будуть імпортуватися всі дані та функції з інших файлів та вводитись нові дані

import os

from StudentList import StudentList

from Utils import Utils

BASE\_DIR = os.path.dirname(\_\_file\_\_)

DEF\_NAME = os.path.join(BASE\_DIR, "lab\_03.csv")

def main():

    student\_list = StudentList()

    # Зчитування даних з файлу

    Utils.import\_data(DEF\_NAME, student\_list)

    while True:

        choice = input("Please specify the action [ C create, U update, D delete, P print,  X exit ] ").lower()

        match choice:

            case "c":

                name = input("Enter student name: ")

                phone = input("Enter student phone: ")

                group = input("Enter student group: ")

                address = input("Enter student address: ")

                student\_list.add\_student(name, phone, group, address)

            case "u":

                name = input("Enter the name of the student to update: ")

                new\_name = input("Enter new name or press Enter to skip: ")

                phone = input("Enter new phone or press Enter to skip: ")

                group = input("Enter new group or press Enter to skip: ")

                address = input("Enter new address or press Enter to skip: ")

                if not student\_list.update\_student(name, new\_name, phone, group, address):

                    print("Student not found.")

            case "d":

                name = input("Enter the name of the student to delete: ")

                if not student\_list.delete\_student(name):

                    print("Student not found.")

            case "p":

                student\_list.print\_all\_students()

            case "x":

                Utils.save\_data(DEF\_NAME, student\_list)

                print("Exiting program.")

                break

            case \_:

                print("Invalid choice.")

main()

Створив файл Student.py який зберігає інформацію про одного студента, включаючи його ім'я, номер телефону, групу та адресу. В цьому файлі створюєтся новий об'єкт класу Student, в якому задаются його властивості (ім'я, телефон, групу, адресу) через конструктор \_\_init\_\_. Метод \_\_str\_\_ дозволяє отримати текстову інформацію про об'єкт у зручному форматі.

class Student:

    def \_\_init\_\_(self, name, phone, group, address):

        self.name = name

        self.phone = phone

        self.group = group

        self.address = address

    def \_\_str\_\_(self):

        return f"Student name: {self.name}, Phone: {self.phone}, Group: {self.group}, Address: {self.address}"

Створив файл StudentList.py який управляє списком об'єктів Student, зберігає список студентів, забезпечуючи впорядкованість за іменем, надає інтерфейс для виконання операцій (створення, читання, оновлення, видалення). В цьому файлі створюється об'єкт класу StudentList, який містить список студентів, методи класу виконують операції над цим списком (додавання, видалення, оновлення, виведення на екран).

from Student import Student

class StudentList:

    def \_\_init\_\_(self):

        self.students = []

    def add\_student(self, name, phone, group, address):

        student = Student(name, phone, group, address)

        self.students.append(student)

        self.students.sort(key=lambda x: x.name)

    def delete\_student(self, name):

        for student in self.students:

            if student.name == name:

                self.students.remove(student)

                return True

        return False

    def update\_student(self, name, new\_name=None, phone=None, group=None, address=None):

        for student in self.students:

            if student.name == name:

                student.name = new\_name or student.name

                student.phone = phone or student.phone

                student.group = group or student.group

                student.address = address or student.address

                self.students.sort(key=lambda x: x.name)

                return True

        return False

    def print\_all\_students(self):

        if not self.students:

            print("The list of students is empty.")

        else:

            for student in self.students:

                print(student)

Створив файл Utils.py який виконує функцію читання даних про студентів із файлу та запис їх назад у файл.

import\_data: Читає файл, створює студентів і додає їх до списку.

save\_data: Зберігає інформацію про студентів зі списку у форматі CSV.

import csv

from Student import Student

class Utils:

    @staticmethod

    def import\_data(file\_name, student\_list):

        try:

            with open(file\_name, 'r', encoding='utf-8') as file:

                reader = csv.DictReader(file)

                for row in reader:

                    student\_list.add\_student(

                        row.get("name", "Unknown"),

                        row.get("phone", "Unknown"),

                        row.get("group", "Unknown"),

                        row.get("address", "Unknown")

                    )

        except FileNotFoundError:

            print(f"File '{file\_name}' not found. Starting with an empty list.")

        except Exception as e:

            print(f"File loading error: {e}")

    @staticmethod

    def save\_data(file\_name, student\_list):

        try:

            with open(file\_name, 'w', encoding='utf-8', newline='') as file:

                fieldnames = ["name", "phone", "group", "address"]

                writer = csv.DictWriter(file, fieldnames=fieldnames)

                writer.writeheader()

                for student in student\_list.students:

                    writer.writerow({

                        "name": student.name,

                        "phone": student.phone,

                        "group": student.group,

                        "address": student.address

                    })

            print(f"Data successfully saved to '{file\_name}'")

        except Exception as e:

            print(f"File saving error: {e}")

**Висновок:** Використовуючи теоретичне підґрунтя про ООП у мові Python я переробив програму телефонного довідника студентів використовуючи принципи ООП для формування відомостей про студентів.