**Лабораторна робота №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 до лабораторної роботи**

1. Завантаження початкових данних для довідника з csv файлу. Приклад csv файлу (lab2.csv), та оригінальній 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**

Для початку я створив функцію printAllList яка виводить на екран усіх студентів, які є у списку students, та перевіряє, чи список студентів порожній. Якщо так, повідомляє користувача. Обробляє список студентів, формуючи текст для кожного студента з його даними, та виводить цей текст на екран.

def printAllList():

    if not students:

        print("The list of students is empty")

        return

    for elem in students:

        strForPrint = (

            "Student name is " + elem["name"] +

            ",  Phone is " + elem["phone"] +

            ",  Group is " + elem["group"] +

            ",  Address is " + elem["address"]

        )

        print(strForPrint)

    return

Потім я створив функцію addNewElement яка створює нового студента при необхідності. Збирає дані студента через запити до користувача, використовує параметри за замовчуванням ("Unknown"), якщо користувач не вводить дані. Вставляє нового студента у список у правильну позицію (за алфавітом), та повідомляє про успішне додавання.

def addNewElement():

    name = input("Please enter student name: ") or "Unknown"

    phone = input("Please enter student phone: ") or "Unknown"

    group = input("Please enter student group: ") or "Unknown"

    address = input("Please enter student address: ") or "Unknown"

    newItem = {"name": name, "phone": phone, "group": group, "address": address}

    insertPosition = 0

    for item in students:

        if name > item["name"]:

            insertPosition += 1

        else:

            break

    students.insert(insertPosition, newItem)

    print("New element has been added")

    return

Створив функцію deleteElement яка видаляє студента з списку

def deleteElement():

    name = input("Please enter name to be deleted: ")

    deletePosition = -1

    for item in students:

        if name == item["name"]:

            deletePosition = students.index(item)

            break

    if deletePosition == -1:

        print("Element was not found")

    else:

        print("Delete position " + str(deletePosition))

        del students[deletePosition]

    return

Створив функцію updateElement яка оновлює інформацію про студента. Запитує ім'я студента, якого потрібно оновити, шукає відповідний запис у списку, та виводить поточну інформацію студента. Пропонує користувачу оновити поля. Якщо користувач не вводить нових даних, зберігаються старі. Якщо ім'я змінюється, студент переміщується на нову позицію у списку.

def updateElement():

    name = input("Please enter name to be updated: ")

    updatePosition = -1

    for item in students:

        if name == item["name"]:

            updatePosition = students.index(item)

            break

    if updatePosition == -1:

        print("Student not found")

    else:

        name1 = students[updatePosition]["name"]

        phone1 = students[updatePosition]["phone"]

        group1 = students[updatePosition]["group"]

        address1 = students[updatePosition]["address"]

        curinf = (

            "Student current information: name — " + name1 +

            " , phone — " + phone1 +

            ", group — " + group1 +

            " , address — " + address1

        )

        print(curinf)

        name = input("Enter new name or press Enter to skip:") or name1

        phone = input("Enter new phone or press Enter to skip:") or phone1

        group = input("Enter new group or press Enter to skip:") or group1

        address = input("Enter new address or press Enter to skip:") or address1

        if name == name1 and phone == phone1 and group == group1 and address == address1:

            print("You haven't updated student information")

        elif name == name1:

            students[updatePosition]["phone"] = phone

            students[updatePosition]["group"] = group

            students[updatePosition]["address"] = address

        else:

            updatedItem = {"name": name, "phone": phone, "group": group, "address": address}

            del students[updatePosition]

            insertPosition = 0

            for item in students:

                if name > item["name"]:

                    insertPosition += 1

            students.insert(insertPosition, updatedItem)

        print("Information has been updated")

    return

Створив функцію import\_data яка завантажує дані студентів з CSV-файлу. Відкриває вказаний файл для читання, обробляє його рядки, додаючи кожного студента до списку students. Якщо файл відсутній, створює порожній список і виводить відповідне повідомлення

def import\_data(file\_name):

    try:

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

            reader = csv.DictReader(file)

            for row in reader:

                students.append({

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

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

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

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

                })

    except FileNotFoundError:

        print(f"File '{file\_name}' not found \nThe initial list will be empty")

    except Exception as e:

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

Створив функцію save\_data яка зберігає оновлену інформацію у CSV-файл. Відкриває файл для запису, записує заголовки колонок (name, phone, group, address). Зберігає кожного студента у файл, та у разі помилки виводить повідомлення.

def save\_data(file\_name):

    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()

            writer.writerows(students)

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

    except Exception as e:

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

Функцію main за допомогою якої виконується навігація по програмі

def main():

    if len(argv) == 1:

        data\_file = DEF\_NAME

    else:

        data\_file = os.path.join(BASE\_DIR, argv[1])  # Используем файл, указанный в аргументах

    print(f"File used '{data\_file}'")

    import\_data(data\_file)

    while True:

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

        match chouse:

            case "C" | "c":

                print("New element will be created:")

                addNewElement()

            case "U" | "u":

                print("Existing element will be updated")

                updateElement()

            case "D" | "d":

                print("Element will be deleted")

                deleteElement()

            case "P" | "p":

                print("List will be printed")

                printAllList()

            case "X" | "x":

                save\_data(data\_file)

                print("Exit()")

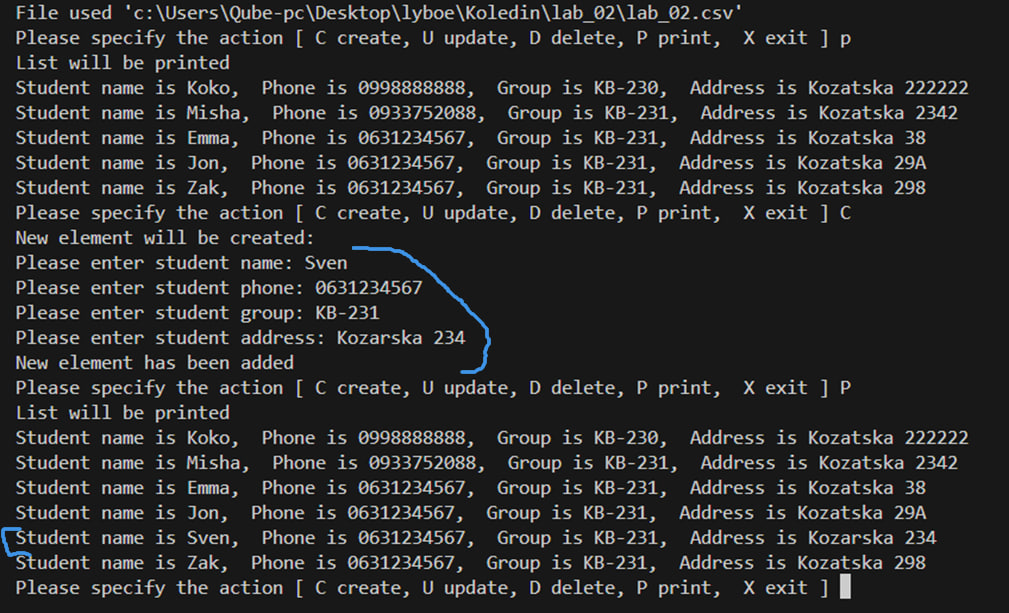
                break

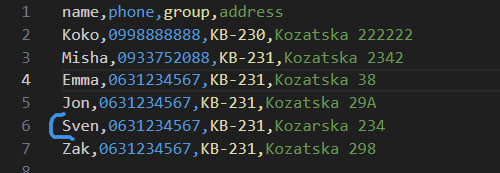
            case \_:

                print("Wrong choice")

Перевірив чи працює код:

Результат:





Текст програми:

import os

import csv

from sys import argv

# Определяем путь к файлу по умолчанию в той же папке, где находится скрипт

BASE\_DIR = os.path.dirname(\_\_file\_\_)  # Получаем директорию текущего файла

DEF\_NAME = os.path.join(BASE\_DIR, "lab\_02.csv")  # lab\_02.csv в папке с программой

students = []

def printAllList():

    if not students:

        print("The list of students is empty")

        return

    for elem in students:

        strForPrint = (

            "Student name is " + elem["name"] +

            ",  Phone is " + elem["phone"] +

            ",  Group is " + elem["group"] +

            ",  Address is " + elem["address"]

        )

        print(strForPrint)

    return

def addNewElement():

    name = input("Please enter student name: ") or "Unknown"

    phone = input("Please enter student phone: ") or "Unknown"

    group = input("Please enter student group: ") or "Unknown"

    address = input("Please enter student address: ") or "Unknown"

    newItem = {"name": name, "phone": phone, "group": group, "address": address}

    insertPosition = 0

    for item in students:

        if name > item["name"]:

            insertPosition += 1

        else:

            break

    students.insert(insertPosition, newItem)

    print("New element has been added")

    return

def deleteElement():

    name = input("Please enter name to be deleted: ")

    deletePosition = -1

    for item in students:

        if name == item["name"]:

            deletePosition = students.index(item)

            break

    if deletePosition == -1:

        print("Element was not found")

    else:

        print("Delete position " + str(deletePosition))

        del students[deletePosition]

    return

def updateElement():

    name = input("Please enter name to be updated: ")

    updatePosition = -1

    for item in students:

        if name == item["name"]:

            updatePosition = students.index(item)

            break

    if updatePosition == -1:

        print("Student not found")

    else:

        name1 = students[updatePosition]["name"]

        phone1 = students[updatePosition]["phone"]

        group1 = students[updatePosition]["group"]

        address1 = students[updatePosition]["address"]

        curinf = (

            "Student current information: name — " + name1 +

            " , phone — " + phone1 +

            ", group — " + group1 +

            " , address — " + address1

        )

        print(curinf)

        name = input("Enter new name or press Enter to skip:") or name1

        phone = input("Enter new phone or press Enter to skip:") or phone1

        group = input("Enter new group or press Enter to skip:") or group1

        address = input("Enter new address or press Enter to skip:") or address1

        if name == name1 and phone == phone1 and group == group1 and address == address1:

            print("You haven't updated student information")

        elif name == name1:

            students[updatePosition]["phone"] = phone

            students[updatePosition]["group"] = group

            students[updatePosition]["address"] = address

        else:

            updatedItem = {"name": name, "phone": phone, "group": group, "address": address}

            del students[updatePosition]

            insertPosition = 0

            for item in students:

                if name > item["name"]:

                    insertPosition += 1

            students.insert(insertPosition, updatedItem)

        print("Information has been updated")

    return

def import\_data(file\_name):

    try:

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

            reader = csv.DictReader(file)

            for row in reader:

                students.append({

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

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

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

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

                })

    except FileNotFoundError:

        print(f"File '{file\_name}' not found \nThe initial list will be empty")

    except Exception as e:

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

def save\_data(file\_name):

    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()

            writer.writerows(students)

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

    except Exception as e:

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

def main():

    if len(argv) == 1:

        data\_file = DEF\_NAME

    else:

        data\_file = os.path.join(BASE\_DIR, argv[1])  # Используем файл, указанный в аргументах

    print(f"File used '{data\_file}'")

    import\_data(data\_file)

    while True:

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

        match chouse:

            case "C" | "c":

                print("New element will be created:")

                addNewElement()

            case "U" | "u":

                print("Existing element will be updated")

                updateElement()

            case "D" | "d":

                print("Element will be deleted")

                deleteElement()

            case "P" | "p":

                print("List will be printed")

                printAllList()

            case "X" | "x":

                save\_data(data\_file)

                print("Exit()")

                break

            case \_:

                print("Wrong choice")

main()

**Висновок:** Використовуючи теоретичне підґрунтя про роботу з файлами та тестування коду у мові Python я розширив програму телефонного довідника студентів додавши функціонал, що був вказаний в завданні до лабораторної роботи