**Звіт про виконання практичних завдань до лекцій з курсу Технології програмування на мові Python**

Звіт до Лабораторної роботи №1

Списки. Словники. Кортежі

**Мета роботи:** Використовуючи теоретичне підґрунтя про складні структури даних Списки, Словники Кортежі, та використовуючи існуючий код, доробити програму додавши функціонал, що буде вказано в завданні до лабораторної роботи.

Під час виконання Лабораторної роботи №1 було надано варіанти рішення до наступних задач:

Реалізувати відсортований телефонний довідник студентів групи.

Для виконання задання надано частину готового функціоналу, яка розміщена в одній директорії з завданням до лабораторної роботи та має назву lab\_01.py.

Частина готового функціоналу реалізує безкінечний цикл запитів до користувача. Типи запитів: додати нового студента, змінити данні про існуючого студента, видалити запис, роздрукувати всю таблицю та вихід із програми. Реалізован функціонал додавання нового запису та видалення існуючого. Всі дії відбуваються з відсортованим списком студентів.

Перед виконанням роботи слід ознайомитись з існуючим функціоналом.

Необхідно розширити відомості про студента до 4х полів. На даний час використовується лише два поля (name та phone).

Необхідно переробити існуючий функціонал враховуючи розширення відомості про студента до 4х полів.

Необхідно реалізувати з нуля функціонал зміни інформації про студента враховуючи той факт, що вже існує реалізація додавання нового запису та видалення існуючого. При зміні інформації про студента список має залишатись відсортованим.

Текст програми разом зі звітом розмістити в каталозі lab\_01. Каталог lab\_01 розмістити в каталозі, що використовується для виконання практичних завдань по кожній лекції та має назву TP-KB-22[1 or 2]-Name-Surname.

**Хід виконання завдання:**

У цьому завданні я реалізував **відсортований** телефонний довідник студентів. За основу я взяв частину готового функціоналу, з лабораторної роботи №1.

Частина готового функціоналу реалізує безкінечний цикл запитів до користувача, таких як: додати нового студента, змінити дані про існуючого студента, видалити запис, роздрукувати всю таблицю та вихід із програми. В зразку вже було реалізовано функціонал додавання нового запису та видалення існуючого. Всі дії відбуваються з відсортованим списком студентів.

Спочатку я розширив відомості про студентів до 4х полів (додав поля: "kurs", "group"). А також переробив існуючий функціонал враховуючи розширення відомостей про студентів до 4х полів.

Далі я реалізував з нуля функціонал зміни інформації про студента у функції «updateElement()» враховуючи той факт, що вже існує реалізація додавання нового запису та видалення існуючого.

**Текст фінкції updateElement():**

def updateElement():

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

    updatePosition = -1

    for item in list:

        if name == item["name"]:

            updatePosition = list.index(item)

            print("Current Student is: ", list[updatePosition])

            break

    if updatePosition == -1:

        print("Element was not found")

        return

    else:

        new\_name = input("Enter new name: ")

        new\_phone = input("Enter new phone: ")

        new\_kurs = input("Enter new kurs: ")

        new\_group = input("Enter new group: ")

        newItem = {"name": new\_name, "phone": new\_phone, "kurs": new\_kurs, "group": new\_group}

    del list[updatePosition]

    insertPosition = 0

    for item in list:

        if new\_name > item["name"]:

            insertPosition += 1

        else:

            break

    list.insert(insertPosition, newItem)

    print("Element has been updated")

    return

**Принцип дії функції updateElement()**

Після виклику функції updateElement() із головної функції main() вводиться ім’я студента, дані якого ми будемо редагувати.

Далі в циклі відбувається пошук позиції (індекса) студента з таким іменем. Якщо студента з таким іменем знайдено, то індекс цього студента запам’ятовується у змінній «updatePosition» для подальшої обробки. Якщо студента з таким іменем немає то виводиться повідомлення про помилку й відбувається повернення до головної функції main().

Далі вводяться нові ім’я студента, телефон, курс, група та заносяться в проміжний словник «newItem».

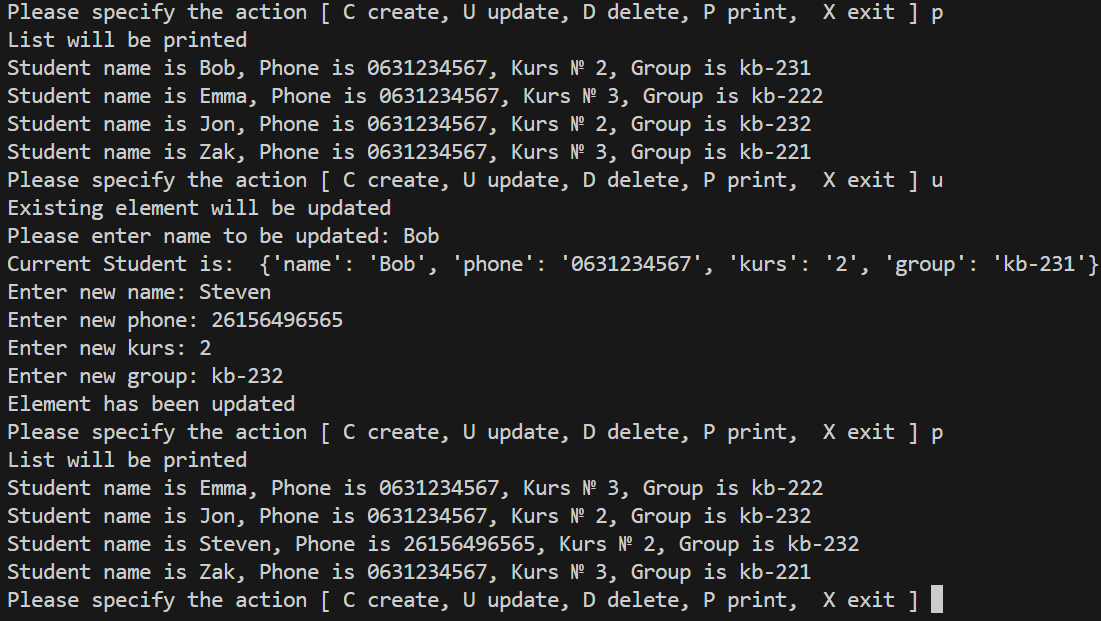
Далі дані про цього студента видаляються зі списку.

Далі в циклі знаходиться позиція (індекс) зміненого імені студента у основному списку, так як список має бути відсортований.

Після знаходження позиції елемента нові дані про студента вставляються в основний список. Таким чином сортування списку зберігається.

Після цього всього виводиться повідомлення про успішно виконану операцію оновлення та програма повертається до основної функції функції main().

Результат виконання програми:



Текст програми:

list = [

    {"name":"Bob", "phone":"0631234567", "kurs":"2", "group":"kb-231"},

    {"name":"Emma", "phone":"0631234567", "kurs":"3", "group":"kb-222"},

    {"name":"Jon", "phone":"0631234567", "kurs":"2", "group":"kb-232"},

    {"name":"Zak", "phone":"0631234567", "kurs":"3", "group":"kb-221"}

]

def printAllList():

    for elem in list:

        strForPrint="Student name is "+elem["name"]+", Phone is "+elem["phone"]+", Kurs в„– "+elem["kurs"]+", Group is "+elem["group"]

        print(strForPrint)

    return

def addNewElement():

    name = input("Pease enter student name: ")

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

    kurs = input("Please enter student kurs: ")

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

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

    insertPosition = 0

    for item in list:

        if name > item["name"]:

            insertPosition += 1

        else:

            break

    list.insert(insertPosition, newItem)

    print("New element has been added")

    return

def deleteElement():

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

    deletePosition = -1

    for item in list:

        if name == item["name"]:

            deletePosition = list.index(item)

            break

    if deletePosition == -1:

        print("Element was not found")

    else:

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

        # list.pop(deletePosition)

        del list[deletePosition]

    return

def updateElement():

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

    updatePosition = -1

    for item in list:

        if name == item["name"]:

            updatePosition = list.index(item)

            print("Current Student is: ", list[updatePosition])

            break

    if updatePosition == -1:

        print("Element was not found")

        return

    else:

        new\_name = input("Enter new name: ")

        new\_phone = input("Enter new phone: ")

        new\_kurs = input("Enter new kurs: ")

        new\_group = input("Enter new group: ")

        newItem = {"name": new\_name, "phone": new\_phone, "kurs": new\_kurs, "group": new\_group}

    del list[updatePosition]

    insertPosition = 0

    for item in list:

        if new\_name > item["name"]:

            insertPosition += 1

        else:

            break

    list.insert(insertPosition, newItem)

    print("Element has been updated")

    return

def main():

    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()

                printAllList()

            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":

                print("Exit()")

                break

            case \_:

                print("Wrong chouse")

main()

Посилання на github:

<https://github.com/diachenkom/TP-KB-231-Maksym-Diachenko>

Знімок екрану з посилання на github:

