**Бобер Євгеній Антонович КБ-241**

**Звіт про виконання 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.

**Теоретичні відомості**

**Списки**

Масив – набір фіксованої кількості елементів, що розміщені в пам’яті комп’ютера безпосередньо один за одним, а доступ до них здійснюється за індексом (номер даного елементу в масиві).

В Python для реалізації масиву використовуються списки. Список – тип даних, що представляє собою послідовність певних значень, що можуть повторюватись. Але на відміну від масиву – кількість елементів у списку може бути довільною.

Списки – структура даних, що може містити елементи різних типів, що перераховані через кому та заключені в квадратні дужки.

**Словники**

Словник дуже схожий на список, але порядок елементів в ньому не має значення, і вони вибираються не за допомогою зміщення. Замість цього для кожного значення вказується пов’язаний з ним унікальний ключ. Таким ключем може бути об’єкт одного з незмінних типів: рядок, булева змінна, ціле число, число з плаваючою точкою, кортеж і іншими об’єктами. Елементи словника можуть містити об’єкти довільного типу даних і мати необмежений рівень вкладеності. Елементи в словниках розташовуються в довільному порядку.

Словники можна змінювати – це означає, що можна додати, видалити і змінити їх елементи, які мають вигляд "ключ – значення"

**Кортежі**

Кортежі, як і списки, є послідовностями довільних елементів. На відміну від списків кортежі незмінні.

Всі операції над списками, що не змінюють список (додавання, множення на число, функції index() і count() і деякі інші операції) можна застосовувати до кортежів. Можна також по-різному змінювати елементи місцями і так далі.

Щоб створити порожній кортеж використовується оператор ().

| xy = (12, 21)  print(xy) |
| --- |

Результат

| (12, 21) |
| --- |

Функція перетворення tuple() створює кортежі з інших об’єктів

| students = ['Alex', 'Helen', 'Olga']  print(students)  tuple\_students = tuple(students)  print(tuple\_students) |
| --- |

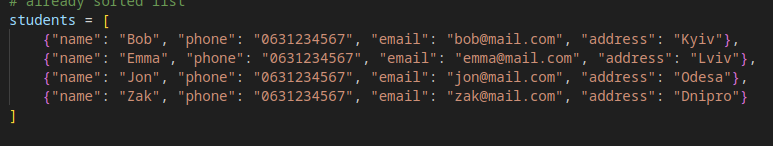
Результат

| ['Alex', 'Helen', 'Olga']  ('Alex', 'Helen', 'Olga') |
| --- |

**Хід роботи**

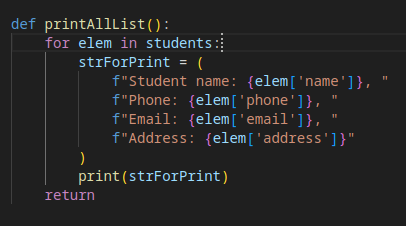
1. Після перенесення файлу лабораторної в робоче середовище я покроково виконую апдейт коду для того щоб наявний функціонал працював для студентів з 4-ма полями.

Для початку давайте визначимо всі 4 поля для кожного студента в списку students:

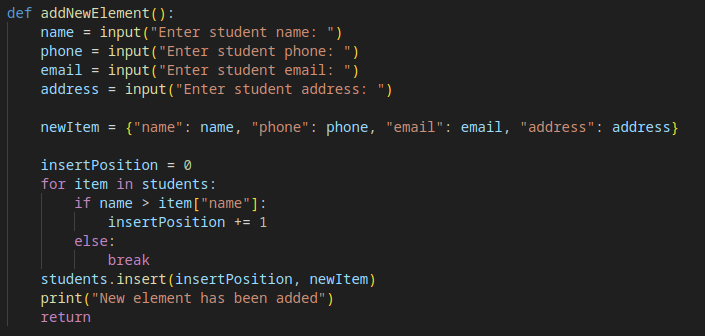


Для прикладу додали значення email та address.

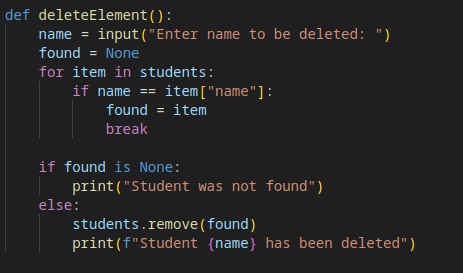
1. Оновлюємо функцію printAllList для того щоб вона виводила всі значення студентів:



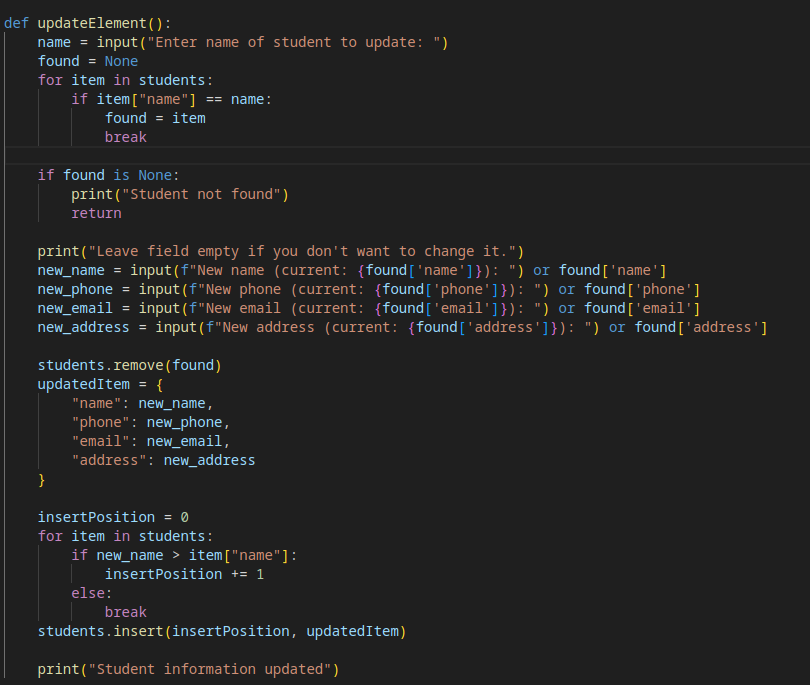
1. Оновлюємо функцію addNewElement, де також будемо приймати нові значення в змінні email та address:



1. Трішки видозмінюємо функцію deleteElement, в якій трішки упрощуємо логіку пошуку елемента списку, і також використовуємо назву списку students замість list, оскільки назва list перекриває вбудований тип Python:



1. Прописуємо функціонал для updateElement:



Шукаємо студента по імені, якщо не знайдено то виходимо із функції. Пропонуємо оновити дані для студента, якщо користувач залишає поле порожнім, то використовується старе значення. Після чого видаляється старий елемент і створюється новий updatedItem. Далі цей елемент вставляється в правильну позицію. За допомогою пошуку по новому імені в списку, ми можемо двигати точку в яку потрібно вставити даний елемент.

1. Головна функція main, в якій працює безкінечний цикл, котрий запитує в користувача яку функцію виконати (C, U, D, P, X).

Виконує відповідну функцію:

* C → додає нового студента
* U → змінює інформацію про існуючого
* D → видаляє
* P → показує список
* X → завершує роботу

В кінці коду ми визиваємо головну функцію main.

**Висновок**

У ході виконання лабораторної роботи було розроблено програму для ведення відсортованого телефонного довідника студентів групи. Доопрацьовано готовий функціонал:

* розширено структуру запису про студента з 2-х до 4-х полів (ім’я, телефон, email, адреса);
* реалізовано додавання, видалення, виведення та зміну інформації про студента;
* забезпечено підтримку відсортованого списку після кожної операції;
* реалізовано інтерфейс взаємодії з користувачем через нескінченний цикл із меню вибору дій.

Отриманий додаток дозволяє ефективно керувати списком студентів, підтримує актуальність даних і гарантує збереження сортування. Під час виконання завдання було закріплено навички роботи зі словниками, списками, циклами та умовними конструкціями у Python.