Лабораторна робота №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.

Списки служать для того, щоб зберігати об’єкти в певному порядку, особливо якщо порядок або вміст можуть змінюватися. Можна змінювати список, додати в нього нові елементи, а також видалити або перезаписати існуючі. Можна змінити кількість елементів у списку, а також самі елементи. Одне і те ж значення може зустрічатися в списку кілька разів.

Приклад визначення списку

```
list_num = ["1", "2", "3"]
print(list_num)
list_str = ["aa", "bb", "cc"]
print(list_str)
```

Результат

```
['1', '2', '3']
['aa', 'bb', 'cc']
```

Крім того, за допомогою функції list() можна створити порожній список

Список містить різні дані, звертатися до яких можна через ім’я списку та вказавши зміщення необхідного елементу

```
students = ["Ihor", "Dima", "Serge"]
print(students)
print(students[0])
```

Результат

```
['Ihor', 'Dima', 'Serge']
Ihor
```

Використовуючи методи списку можна виконувати необхідні операції. Для додавання елементів в кінець списку – використовують метод **append()**. Можна об’єднати один список з іншим за допомогою методу **extend()**. Функція **append()** додає елементи тільки в кінець списку. Коли потрібно додати елемент в задану позицію, використовується функція **insert().**

За допомогою функції **pop()** можна отримати елемент зі списку і в той же час видалити його. Якщо викликати функцію **pop()** і вказати зсув, вона поверне елемент, що знаходиться в заданій позиції. Якщо аргумент не вказано – буде використано значення -1. Так, виклик pop(0) поверне головний (початковий) елемент списку, а виклик pop() або pop(-1) – кінцевий елемент.

Для проходження по списку використовуються цикли

```
students = ["Ihor", "Dima", "Serge"]
for name in students:
    print(name)
```

Результат

```
Ihor
Dima
Serge
```

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"

Щоб створити словник, необхідно заключити в фігурні дужки ({}) розділені комами пари ключ: значення.

```
animals = {
    "dog" : 4,
    "cat" : 4,
    "goose": 2
}

print(animals)
```

Результат

```
{'dog': 4, 'cat': 4, 'goose': 2}
```

Можна використовувати функцію **dict**(), щоб створити порожній словник, якщо не вказати параметри функції

Звернення до елементів словника здійснюється за допомогою квадратних дужок, в яких вказується ключ.

```
animals = {
    "dog" : 4,
    "cat" : 4,
    "goose": 2
}
print(animals["dog"])
```

Результат

```
4
```

Щоб дізнатися, чи міститься в словнику якийсь ключ, використовується ключове слово **in**. Якщо ключ знайдений, то повертається значення True, в іншому випадку – **False**.

```
animals = {
    "dog" : 4,
    "cat" : 4,
    "goose": 2
}

print("cat" in animals)
```

Оскільки словники відносяться до змінюваних типів даних, то можна додати або змінити елемент по ключу. Додати елемент в словник досить легко. Потрібно просто звернутися до елементу по його ключу і привласнити йому значення. Якщо ключ вже існує в словнику, наявне значення буде замінено новим. Якщо ключ новий, він і вказане значення будуть додані в словник.

Для словників розроблено набір методів. **update()** – додає елементи в словник. Метод змінює поточний словник і нічого не повертає.

Видалити елемент зі словника можна за допомогою інструкції **del**.

```
dict_2 = {"a": 1, "b": 2}
print(dict_2)
del dict_2 ["b"] # Видаляємо елемент з ключем "b"
print(dict_2)
```

Результат:

```
{'a': 1, 'b': 2}
{'a': 1}
```

Щоб видалити всі ключі і значення зі словника, слід використовувати функцію **clear()** або просто привласнити порожній словник заданому імені.

Скориставшись функцією **keys()** можна отримати всі ключі словника. Щоб отримати всі значення словника, використовується функція **values().** Щоб

отримати всі пари "ключ – значення" із словника, використовується функція **items**().

```
testDisct = {"a": 1, "b": 2}
print(testDisct.keys())
print(testDisct.values())
print(testDisct.items())
```

Результат:

```
dict_keys(['a', 'b'])
dict_values([1, 2])
dict_items([('a', 1), ('b', 2)])
```

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

Завдання до лабораторної роботи

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. Реалізований 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).

```python
students = [
    {"name": "Bob",  "phone": "0631234567", "email": "bob@gmail.com",  "address": "Kyiv"},
    {"name": "Emma", "phone": "0632345678", "email": "emma@gmail.com", "address": "Lviv"},
    {"name": "Jon",  "phone": "0633456789", "email": "jon@gmail.com",  "address": "Odesa"},
    {"name": "Zak",  "phone": "0634567890", "email": "zak@gmail.com",  "address": "Kharkiv"}
]

def printAllList():
    print("\nСПИСОК СТУДЕНТІВ:")
    for elem in students:
        print(f"{elem['name']}: {elem['phone']}, {elem['email']}, {elem['address']}")
    print()
    return

def addNewElement():
    print("\n=== Додавання нового студента ===")
    name = input("Введіть ім'я студента: ")
    phone = input("Введіть телефон: ")
    email = input("Введіть email: ")
    address = input("Введіть адресу: ")

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

    insertPosition = 0
    for item in students:
        if name > item["name"]:
            insertPosition += 1
        else:
            break
    students.insert(insertPosition, newItem)

    print("Новий запис додано!\n")
    return

def deleteElement():
    print("\n=== Видалення студента ===")
    name = input("Введіть ім'я студента для видалення: ")

    deletePosition = -1
    for item in students:
        if name == item["name"]:
            deletePosition = students.index(item)
            break

    if deletePosition == -1:
        print("Студента не знайдено.\n")
    else:
        del students[deletePosition]
```

```
            del students[deletePosition]
            print(f"Студента '{name}' видалено.\n")
        return

    def updateElement():
        print("\n=== Оновлення інформації ===")
        name = input("Введіть ім'я студента, якого потрібно оновити: ")

        updatePosition = -1
        for item in students:
            if name == item["name"]:
                updatePosition = students.index(item)
                break

        if updatePosition == -1:
            print("Студента не знайдено.\n")
            return

        current = students[updatePosition]
        print("Введіть нові дані (Enter — щоб залишити старе значення):")

        new_name = input(f"Ім'я [{current['name']}]: ") or current['name']
        new_phone = input(f"Телефон [{current['phone']}]: ") or current['phone']
        new_email = input(f"Email [{current['email']}]: ") or current['email']
        new_address = input(f"Адреса [{current['address']}]: ") or current['address']

        updated_item = {
            "name": new_name,
            "phone": new_phone,
            "email": new_email,
            "address": new_address
        }

        del students[updatePosition]

        insertPosition = 0
        for item in students:
            if new_name > item["name"]:
                insertPosition += 1
            else:
                break
        students.insert(insertPosition, updated_item)

        print("Інформацію оновлено!\n")
        return
```

```
        print("Інформацію оновлено!\n")
        return

    def main():
        while True:
            choice = input("Оберіть дію [C - додати, U - оновити, D - видалити, P - показати, X - вихід]: ")

            match choice.lower():
                case "c":
                    addNewElement()
                    printAllList()
                case "u":
                    updateElement()
                    printAllList()
                case "d":
                    deleteElement()
                    printAllList()
                case "p":
                    printAllList()
                case "x":
                    print("Програму завершено.")
                    break
                case _:
                    print("Невірна команда, спробуйте ще раз.\n")

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

Висновок: Я навчився використовувати теоретичне підґрунтя про складні структури даних Списки, Словники Кортежі, та використовуючи

існуючий код, доробив програму додавши функціонал, що був вказаний в завданні до лабораторної роботи.