Структури даних: списки, кортежі, словники

GENIUS space

Списки

Створення списку:

Список в Python - це змінна, яка може містити інші об'єкти. Список створюється за допомогою квадратних дужок [] або функції list(), і об'єкти в списку розділяються комами.

```
my_list = [1, 2, 3, 4, 5]  # Створення списку чисел
fruits = ["apple", "banana", "cherry"]  # Створення списку рядків
mixed_list = [1, "apple", True, 3.14]  # Створення списку з різними типами даних
empty_list = []  # Порожній список
```

Індексація списку:

Елементи списку індексуються, починаючи з 0 для першого елемента, 1 для другого і так далі. Щоб отримати доступ до елемента за його індексом, використовуйте квадратні дужки і індекс:

```
first_element = my_list[0]  # Отримання першого елемента
second_element = my_list[1]  # Отримання другого елемента
```

Зрізи списків (slicing):

Зрізи дозволяють отримувати підсписки списку. Синтаксис для зрізів виглядає так: **my_list[start : stop : step]**, де **start** - індекс початку зрізу, **stop** - індекс кінця зрізу (не включаючи його), **step** - крок.

```
sublist = my_list[1:4]  # Отримання підсписку з другого до четвертого елемента
every_other = my_list[::2]  # Отримання кожного другого елемента
```

GENIUS space

Методи списку

.append(item): Додає **item** в кінець списку.

.insert(index, item): Вставляє **item** на певний **index**.

.remove(item): Видаляє перше входження **item** зі списку.

.pop(): Видаляє і повертає останній елемент списку (або з певного індексу, якщо передати індекс).

.index(item): Повертає індекс першого входження item у списку.

.count(item): Повертає кількість входжень **item** у списку.

.sort(): Сортує список за зростанням.

.reverse(): Розвертає список в зворотньому порядку.

GENIUS space

Кортежі

Створення кортежу:

Кортеж в Python - це колекція об'єктів, які мають фіксовану послідовність і не можуть бути змінені після створення. Кортежі створюються, як і списки, за допомогою круглих дужок **()** або tuple(), і об'єкти в кортежі розділяються комами.

```
my_tuple = (1, 2, 3, 4, 5)  # Створення кортежу чисел
fruits_tuple = ("apple", "banana", "cherry")  # Створення кортежу рядків
mixed_tuple = (1, "apple", True, 3.14)  # Створення кортежу з різними типами даних
single_item_tuple = (42,)  # Створення кортежу з одним елементом
```

Індексація кортежу:

Елементи кортежу індексуються так само, як і елементи списку, починаючи з 0. Щоб отримати доступ до елемента за його індексом, використовуйте квадратні дужки і індекс:

```
first_element = my_tuple[0]  # Отримання першого елемента кортежу
second_element = my_tuple[1]  # Отримання другого елемента кортежу
```

Зрізи кортежів (slicing):

Зрізи кортежів працюють аналогічно до списків. Ви можете отримати підписок кортежу, використовуючи синтаксис **my_tuple[start:stop:step]**.

```
subtuple = my_tuple[1:4]  # Отримання підсписку з другого до четвертого елемента
every_other = my_tuple[::2]  # Отримання кожного другого елемента
```

Кортежі є незмінними:

Одна з основних різниць між кортежами і списками полягає в тому, що кортежі незмінні. Це означає, що після створення кортежу ви не можете змінити його елементи. Наприклад, такий код призведе до помилки:

```
my_tuple[0] = 10  # Помилка! Кортеж незмінний.
```

GENIUS space

Методи кортежу

Кортежі не мають багатьох методів порівняно із списками, оскільки вони незмінні. Однак вони підтримують тільки два методи: **.count()** і **.index()**:

.count(item): Повертає кількість входжень **item** у кортеж.
.index(item): Повертає індекс першого входження **item** у кортеж.

```
count = my_tuple.count(3)  # Підрахунок кількості трійок у кортежі
index = my_tuple.index(4)  # Знаходження індексу першої четвірки
```

GENIUS space

Словники

Створення словника:

Словник в Python - це колекція пар ключ-значення, де ключі унікальні та незмінні, а значення можуть бути будь-якого типу даних. Словники створюються за допомогою фігурних дужок {} або dict(), а пари ключ-значення розділяються двокрапкою.

```
my_dict = {'name': 'John', 'age': 30, 'city': 'New York'}  # Створення словника
empty_dict = {}  # Порожній словник
```

Доступ до значень за ключем:

Ви можете отримати доступ до значень в словнику, вказавши ключ у квадратних дужках або за допомогою методу .get().

```
name = my_dct['name']  # Отримання значення за ключем 'name'
age = my_dict.get('age')  # Отримання значення за ключем 'age' за допомогою методу .get()
```

Зміна значень за ключем:

Значення в словнику можна змінювати, вказавши нове значення за існуючим ключем:

```
my_dict['age'] = 31  # Зміна значення 'age' на 31
```

Додавання нових пар ключ-значення:

Для додавання нової пари ключ-значення просто вказуйте новий ключ та відповідне значення:

```
my_dict['country'] = 'USA'  # Додавання нового ключа 'country' зі значенням ‘USA'
```

Видалення пар ключ-значення:

Ви можете видалити пару ключ-значення за ключем за допомогою оператора del або методу .pop():

```
del my_dict['city']  # Видалення ключа 'city' із словника
country = my_dict.pop('country')  # Видалення та отримання значення за ключем 'country'
```

GENIUS space

Методи словника

clear(): Видаляє всі пари ключ-значення з словника і робить його порожнім.

```
my_dict = {'name': 'John', 'age': 30}
my_dict.clear()  # Зараз my_dict порожній словник {}
```

copy(): Створює копію словника.

```
original_dict = {'name': 'John', 'age': 30}
new_dict = original_dict.copy() # Створить копію original_dict, яка буде присвоєна змінній new_dict
```

get(key, default=None): Повертає значення, якщо ключ існує в словнику, інакше повертає значення за замовчуванням (default, за замовчуванням None).

```
age = my_dict.get('age')  # Повертає 30
country = my_dict.get('country', 'USA')  # Повертає 'USA', оскільки ключ 'country' відсутній
```

items(): Повертає усі пари ключ-значення словника у вигляді кортежів.

```
my_dict = {'name': 'John', 'age': 30}
items = my_dict.items()  # Повертає [('name', 'John'), ('age', 30)]
```

keys(): Повертає список усіх ключів у словнику.

```
my_dict = {'name': 'John', 'age': 30}
keys = my_dict.keys()  # Повертає ['name', ‘age']
```

values(): Повертає список усіх значень у словнику.

```
my_dict = {'name': 'John', 'age': 30}
values = my_dict.values()  # Повертає ['John', 30]
```

pop(key, default=None): Видаляє і повертає значення, що відповідає ключу. Якщо ключ не існує і вказано значення за замовчуванням, то повертає значення за замовчуванням.

```
age = my_dict.pop('age')  # Видаляє ключ 'age' і повертає 30
country = my_dict.pop('country', 'USA')  # Не видаляє ключ 'country' і повертає ‘USA'
```

update(iterable): Оновлює словник, додаючи пари ключ-значення із іншого ітерабельного об'єкту (зазвичай, іншого словника).

```
my_dict = {'name': 'John', 'age': 30}
my_dict.update({'city': 'New York', 'country': 'USA'})  # Оновлює словник з новими парами
```

GENIUS space