

програмування

ТЕМА 7. КОРТЕЖІ

Кортежі

Кортежі є сукупностями з визначеної кількості різнотипних полів.

Кортежі використовують тоді, коли декілька характеристик описують один об'єкт.

- Наприклад, точка простору, відомості про автомобіль, дані працівника тощо.

У мовах програмування кортежі часто називають записами або структурами.

У Python кортежі, як і рядки та на відміну від списків, є такими, що не змінюються (immutable).

Тип кортежу також належить до типів послідовностей.

Носій типу кортеж

Кортеж позначається включенням його елементів у круглі дужки через кому.

(x_1, \dots, x_n)

Нехай множини M_1, \dots, M_n є носіями типів t_1, \dots, t_n , до яких належать x_1, \dots, x_n .

Тоді носієм типу кортежу буде декартів добуток множин M_1, \dots, M_n .

$$M_t = M_1 \times \dots \times M_n$$

Основні операції для кортежів

Операція	Опис
(x_1, \dots, x_n)	Створити кортеж з елементів x_1, \dots, x_n
$()$	Порожній кортеж
$(x,)$	Кортеж з 1 елемента x
<code>tuple(x)</code>	Перетворення x у кортеж (x повинно належати типу, що ітерується)
$s + t$	конкатенація s та t
$s * n$ або $n * s$	n зчеплених копій s
$s[i]$	i -й елемент s , починаючи з 0, якщо $i < 0$, то повертає $(-i)$ елемент з кінця кортежу
$s[i:j]$	Вирізка з s від i до j (підкортеж, що починається з i -го елементу та закінчується $j - 1$ елементом)
$s[i:j:k]$	Вирізка з s від i до j з кроком k
<code>len(s)</code>	довжина s

Основні операції для кортежів.2

Операція	Опис
<code>min(s)</code>	Найменший елемент кортежу s
<code>max(s)</code>	Найбільший елемент кортежу s
<code>s.index(x[, i[, j]])</code>	Індекс першого входження x до s (починаючи з індекса i та перед індексом j)
<code>s.count(x)</code>	Кількість входжень x до s

Відношення для кортежів

Для кортежів визначено 6 стандартних відношень з множини $Rel = \{==, !=, >, <, >=, <= \}$.

Відношення $a == b$ означає попарну рівність всіх елементів двох кортежів a, b .

Відношення $a < b$ визначається рекурсивно:

1. Якщо $a == ()$, $b != ()$, то $a < b == \text{True}$
2. Якщо $b == ()$, то $a < b == \text{False}$
3. Якщо $a != ()$, $b != ()$, $a[0] != b[0]$ то $a < b \equiv a[0] < b[0]$
4. Якщо $a != ()$, $b != ()$, $a[0] == b[0]$ то $a < b \equiv a[1:] < b[1:]$

Інші відношення з множини Rel визначається через бульові операції та відношення $==$ та $<$.

Для обчислення відношень з множини Rel відповідні елементи 2 кортежів повинні мати порівнювані типи (наприклад, обидва числові або обидва рядки і т.д.)

Обчислення відношень для непорівнюваних типів дає помилку.

Відношення для кортежів .2

Окрім відношень з множини *Rel*, для кортежів визначено відношення:

$x \text{ in } a$, $x \text{ not in } a$

- де x – елемент, a – кортеж.

$x \text{ in } a == \text{True}$, коли x входить у a

$x \text{ not in } a == \text{True}$, коли x не входить у a

Інструкції для кортежів

Для кортежів визначено присвоєння та виведення.

```
a = e, print(a)
```

Введення не визначено, тому треба вводити кортеж поелементно.

Визначено також цикл по всіх елементах кортежу

```
for x in a:
```

P

Пакування та розпакування кортежів

Пакування кортежу – це присвоєння виду:

```
t = (x, y)
```

- тобто, створення кортежу з декількох змінних або виразів.

Розпакування кортежу – це обернене присвоєння. Наприклад:

```
x, y = t
```

- де t – кортеж з 2 полів.

У викликах функцій можна також використовувати синтаксис розпакування з «зірочкою» - *t.

У цьому випадку замість кортежу або списку на вхід функції подаються його елементи. Наприклад:

```
d = (1, 2)
```

```
print(*d)
```

покаже окремі поля кортежу 1 2

Приклад

Обчислити координати центру мас системи матеріальних точок простору з одиничною масою та найбільшу відстань між точками

“Паралельне” присвоєння з використанням кортежів

Оскільки поля кортежів можуть бути вказані у лівій та правій частині присвоєння, то допустимі такі присвоєння:

$$a, b = c, d$$

Це присвоєння означає що кортежу (a, b) буде присвоєно значення кортежу (c, d).

Таким чином, можемо для відомої задачі обміну значень 2 змінних написати присвоєння

$$x, y = y, x$$

без явного використання додаткових змінних.

Подібний підхід застосовують при обчисленні елементів послідовностей, заданих рекурентними співвідношеннями вищих порядків.

Приклад

Обчислення заданого числа Фібоначчі

Функції zip, sum та map

Три вбудованих функції `zip()`, `sum()` та `map()` використовують для роботи з послідовностями, в тому числі, з кортежами.

Якщо a , b – послідовності, то `zip(a,b)` утворює послідовність кортежів, які складаються з відповідних елементів a та b .

Тобто, якщо послідовність a складається з елементів a_1, \dots, a_n , послідовність b складається з елементів b_1, \dots, b_n , то `zip(a,b)` складається з елементів $(a_1, b_1), \dots, (a_n, b_n)$.

Кількість аргументів функції `zip` може бути і більшою, ніж 2. У цьому випадку кортежі будуть складатись із більшої кількості полів, рівної кількості аргументів `zip`.

Функції zip, sum та map.2

Часто zip використовують з параметром із зірочкою, що передбачає розпакування аргументу zip(*t), наприклад, у випадку списку, що складається з кортежів.

Це дає можливість не вказувати явно кількість послідовностей, що є аргументами zip.

Функція zip повертає спеціальний об'єкт типу, що ітерується. Тобто, результат zip можна використовувати у циклі for.

Наприклад:

```
for x, y in zip(a, b):  
    print(x, y)
```

Для того, щоб зробити результат zip списком, треба застосувати list(zip(a,b)); кортежем, - tuple(zip(a,b)).

Функція sum

Функція $\text{sum}()$ обчислює суму всіх елементів послідовності a .

Тобто, якщо послідовність a складається з елементів a_1, \dots, a_n , то

$$\text{sum}(a) = a_1 + \dots + a_n.$$

Функція map

Функція map застосовує задану функцію f до всіх елементів послідовності a (якщо f має 1 аргумент).

$\text{map}(f,a)$ повертає послідовність значень функції f від всіх елементів a .

Тобто, якщо послідовність a складається з елементів a_1, \dots, a_n , то $\text{map}(f,a)$ складається з елементів $f(a_1), \dots, f(a_n)$.

$\text{map}(\text{abs}, a)$ повертає послідовність модулів елементів a .

Якщо функція f залежить від більшої кількості аргументів, то у map треба вказати більшу кількість послідовностей, рівну, кількості аргументів f .

Наприклад, якщо f залежить від 3 аргументів, то треба вказувати $\text{map}(f,a,b,c)$, де a, b, c – послідовності.

Для map вірно все, що було описано для zip стосовно розпакування, використання у циклах, результату у вигляді списку або кортежу.

Приклади

Дано матрицю $m \times n$ з дійсних чисел. Обчислити мінімальний елемент матриці $m \times n$.

Дано матрицю $m \times n$ з дійсних чисел. Обчислити транспоновану матрицю $n \times m$.

Іменовані кортежі

Іменовані кортежі (namedtuples) дозволяють звертатися до полів не за їх номером `a[0]`, `a[1]`, ..., а за іменами.

Для того, щоб використовувати іменовані кортежі, треба написати

```
from collections import namedtuple
```

Для того, щоб описати специфічний тип іменованого кортежу, треба написати

```
t = namedtuple('t', ['f1', ..., 'fn'])
```

Іменовані кортежі.2

Наприклад, для точки площини

```
Point2 = namedtuple('Point2', ['x', 'y'])
```

Після цього можемо створити кортеж

```
p = Point2(1,0)
```

та звертатися до полів кортежу за іменами

```
m = p.x
```

Приклад

Обчислити координати центру мас системи матеріальних точок простору з одиничною масою та найбільшу відстань між точками (версія 2)

Резюме

Ми розглянули:

1. Кортежі. Носій для кортежів.
2. Операції, відношення та інструкції для кортежів.
3. Пакування та розпакування
4. Функції zip, sum та map.
5. Іменовані кортежі.

Де прочитати

1. Обвінцев О.В. Інформатика та програмування. Курс на основі Python. Матеріали лекцій. – К., Основа, 2017
2. A Byte of Python (Russian) Версія 2.01 Swaroop С Н (Translated by Vladimir Smolyar),
<http://wombat.org.ua/AByteOfPython/AByteofPythonRussian-2.01.pdf>
3. Марк Лутц, Изучаем Python, 4-е издание, 2010, Символ-Плюс
4. Python 3.4.3 documentation
5. Бублик В.В., Личман В.В., Обвінцев О.В.. Інформатика та програмування. Електронний конспект лекцій, 2003 р.
6. <https://bradmontgomery.net/blog/pythons-zip-map-and-lambda/>