BestProg

PYTHON, ТЕОРИЯ

PYTHON. ОПЕРАЦИИ НАД КОРТЕЖАМИ. ОБХОД КОРТЕЖА. МЕТОДЫ РАБОТЫ С КОРТЕЖАМИ.

15 АПРЕЛЯ, 2020 | BESTPROGISCH

Операции над кортежами. Обход кортежа. Методы работы с кортежами

Содержание

Поиск на других ресурсах:

1. Операция присваивания кортежа =. Примеры

Любому объекту с именем можно присвоить кортеж. Примеры присваивания кортежей.

```
a = () # пустой кортеж
b = (5,'Hello', True) # кортеж из 3-х элементов различных типов
c = ('world', (2.88, "bestprog")) # вложенный кортеж
d = 5, 'Hello', True # такой же кортеж как и b
e = (3.88,) # Кортеж строго из одного элемента
```

⇑

2. Операция tuple() – создание кортежа из итерированного объекта

Кортеж можно создать с помощью операции tuple(). Эта операция принимает параметром итерированный объект, которым может быть другой кортеж, список, строка.

Например

```
# Операция tuple()
# 1. Создание кортежа из слова 'Hello'
d = tuple('Hello'); # d = ('H', 'e', 'l', 'l', 'o')

# 2. Создание кортежа из списка
# Заданный список
lst = [2, "abc", 3.88]

# Создать кортеж
e = tuple(lst) # e = (2, 'abc', 3.88)

# 3. Создание кортежа из другого кортежа
f = tuple((3, 2, 0, -5)) # f = (3, 2, 0, -5)
```

⇑

3. Операция T[*i*]. Получить элемент по индексу

Элемент кортежа может быть получен по индексу.

Пример.

```
# Кортежи. Операция взятия индекса
# 1. Кортеж, содержащий строки
a = ('a', 'bc', 'def', 'ghij')
item = a[2] # item = 'def'

# 2. Кортеж, содержащий другой кортеж и его элементы
b = (a, a[1], True)
item = b[0] # item = ('a', 'bc', 'def', 'ghij')
item = b[1] # item = 'bc'

# 3. Кортеж, содержащий список
c = (1, [2, 3, 4], "text")

# 3.1. Вытянуть список из кортежа
item = c[1] # item = [2, 3, 4], c =   (1, [2, 3, 4], 'text')

# 3.2. Список в кортеже изменяемый (mutable), поэтому его
#      можно изменить
c[1][1] = 8 # c = (1, [2, 8, 4], 'text')
```

⇑

4. Операция T[*i*][*j*]. Получить доступ к составному элементу в кортеже

Как известно, кортеж может содержать вложенные объекты, которыми могут быть списки, строки ли даже другие кортежи. Чтобы получить элемент вложенного объекта нужно использовать операцию взятия индекса

```
T[i][j]
```

где

- ***T*** – имя кортежа;
- ***i*** – позиция вложенного объекта в кортеже;
- ***j*** – позиция элемента во вложенном объекте.

Пример.

```
# Операция T[i][j] - получить элемент по индексу
# 1. Кортеж, который содержит список
A = ( 2.5, ['a', True, 3.17], 8, False, 'z')
item1 = A[1]   # item1 =   ['a', True, 3.17]
item2 = A[1][2] # item2 = 3.17

# 2. Кортеж, который содержит строку
B = ( "Hello", "abcd", 2.55)
item1 = B[0]   # item1 =   'Hello'
item2 = B[0][4] # item2 = 'o'

# 3. Кортеж, который содержит другой кортеж, список, строку
C = ( (1, 2, 5), [2, 7, -100], "Python")
item1 = C[0]   # item1 =   (1, 2, 5)
item2 = C[0][2] # item2 = 5

# 4. Три уровня вложения
D = ( [ 7, True, ('a', 'b', 'cd')], 12, "bestprog")
item1 = D[0]       # item1 =   [7, True, ('a', 'b', 'cd')]
item2 = D[0][2]   # item2 =   ('a', 'b', 'cd')
item3 = D[0][2][1] # item2 = 'b'
```

⇑

5. Операция T[i:j]. Взятие среза в кортеже

С помощью операции взятия среза можно получить другой кортеж. Общая форма операции взятия среза для кортежа следующая

```
T2 = T1[i:j]
```

здесь

- *T2* – новый кортеж, который получается из кортежа *T1*;
- *T1* – исходный кортеж, для которого происходит срез;
- *i*, *j* – соответственно нижняя и верхняя границы среза. Фактически берутся ко вниманию элементы, лежащие на позициях *i*, *i*+1, ..., *j*-1. Значение *j* определяет позицию за последним элементом среза.

Операция взятия среза для кортежа может иметь модификации. Более подробно об использовании срезов в Python описывается в теме:

- **2.3. Доступ по индексам. Срезы. Получение фрагмента строки. Примеры**

Пример.

```
# Операция [i:j] - взятие среза
# 1. Кортеж, содержащий целые числа
A = ( 0, 1, 2, 3)
item = A[0:2]   # item =   (0, 1)

# 2. Кортеж, содержащий список
A = ( 2.5, ['abcd', True, 3.1415], 8, False, 'z')
item = A[1:3] # item = (['abcd', True, 3.1415], 8)

# 3. Кортеж, содержащий вложенный кортеж
A = (3, 8, -11, "program")
B = ("Python", A, True)
item = B[:3] # item = ('Python', (3, 8, -11, 'program'), True)
item = B[1:] # item = ((3, 8, -11, 'program'), True)
```

⇑

6. Конкатенация +

Для кортежей можно выполнять операцию конкатенации, которая обозначается символом +. В простейшем случае для конкатенации двух кортежей общая форма операции следующая

```
T3 = T1 + T2
```

где

- *T1*, *T2* – кортежи, для которых нужно выполнить операцию конкатенации. Операнды *T1*, *T2* обязательно должны быть кортежами. При выполнении операции конкатенации для кортежей, использовать в качестве операндов любые другие типы (строки, списки) запрещено;
- *T3* – кортеж, который есть результатом.

Пример.

```
# Кортежи. Конкатенация +
# Конкатенация двух кортежей
A = (1, 2, 3)
B = (4, 5, 6)
C = A + B # C =   (1, 2, 3, 4, 5, 6)

# Конкатенация кортежей со сложными объектами
D = (3, "abc") + (-7.22, ['a', 5]) # D = (3, 'abc', -7.22, ['a', 5])

# Конкатенация трех кортежей
A = ('a', 'aa', 'aaa')
B = A + (1, 2) + (True, False) # B = ('a', 'aa', 'aaa', 1, 2, True, False)
```

⇑

7. Повторение *

Кортеж может быть образован путем операции повторения, обозначаемой символом *. При использовании в выражении общая форма операции следующая

```
T2 = T1 * n
```

здесь

- *T2* – результирующий кортеж;
- *T1* – исходный кортеж, который нужно повторить *n* раз;
- *n* – количество повторений кортежа *T1*.

Пример.

```
# Кортежи. Повторение *
# Кортеж, который содержит простые числа
A = (1, 2, 3) * 3 # A = (1, 2, 3, 1, 2, 3, 1, 2, 3)

# Кортеж, который содержит вложенные объекты
B = ("ab", ["1", "12"])*2 # A=('ab', ['1','12'], 'ab', ['1','12'])
```

⇑

8. Обход кортежа в цикле. Пример

Элементы кортежа можно последовательно просмотреть с помощью операторов цикла while или for.

Пример

```
# Обход кортежа в цикле
# 1. Цикл for
# Заданный кортеж
A = ("abc", "abcd", "bcd", "cde")

# Вывести все элементы кортежа
for item in A:
    print(item)

# 2. Цикл while
# Исходный кортеж - целые числа
A = (-1, 3, -8, 12, -20)

# Вычислить количество положительных чисел
i = 0
k = 0 # количество положительных чисел

while i < len(A):
    if (A[i]<0):
        k = k + 1
    i = i + 1

# Вывести результат
print("k = ", k)

# 3. Обход в цикле for
# Заданный кортеж, содержащий строки
A = ("abc", "ad", "bcd")

# Сформировать новый список из элементов кортежа A,
# в новом списке B, каждый элемент удваивается
B = [item * 2 for item in A]

print("A = ", A)
print("B = ", B)
```

Результат выполнения программы

```
abc
abcd
bcd
cde
k = 3
A = ('abc', 'ad', 'bcd')
B = ['abcabc', 'adad', 'bcdbcd']
```

⇑

9. Операция in. Проверка вхождения элемента в кортеж

```
# Проверка вхождения элемента в кортеж
# Оператор in
# Заданный кортеж, который содержит строки
A = ("abc", "abcd", "bcd", "cde")

# Ввести элемент
item = str(input("s = "))

if (item in A):
    print(item, " in ", A, " = True")
else:
    print(item, " in ", A, " = False")
```

Результат выполнения программы

```
s = abc
abc in   ('abc', 'abcd', 'bcd', 'cde') = True
```

⇑

10. Методы работы с кортежами

10.1. Метод index(). Поиск позиции элемента в кортеже

Чтобы получить индекс (позицию) элемента в кортеже, нужно использовать метод index(). Общая форма вызова метода следующая

```
pos = T.index(item)
```

где

- *T* – кортеж, в котором осуществляется поиск;
- *pos* – позиция (индекс) элемента *item* в кортеже. Первому элементу соответствует позиция 0. Если элемента нет в кортеже, генерируется исключительная ситуация. Поэтому, перед использованием метода index() рекомендуется делать проверку на наличие элемента (с помощью операции in).

Пример. В примере, в перечне названий дней недели вычисляется порядковый номер дня.

```
# Метод index - определяет позицию (индекс) элемента в кортеже
# Заданный кортеж
A = ("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat")

# Запрос к вводу названия дня недели
day = str(input("Enter day: "))

# Корректно вычислить индекс
if day in A: # проверка, есть ли строка day в кортеже A
    num = A.index(day)
    print("Number of day = ", num + 1)
else:
    num = -1
    print("Wrong day.")
```

Результат работы программы

```
d = 1
```

⇑

10.2. Метод count(). Количество вхождений элемента в кортеж

Чтобы определить количество вхождений заданного элемента в кортеж используется метод count, общая форма которого следующая:

```
k = T.count(item)
```

здесь

- *T* – исходный кортеж;
- *k* – результат (количество элементов);
- *item* – элемент, количество вхождений которого нужно определить. Элемент может быть составным (строка, список, кортеж).

```
# Метод count - подсчет количества вхождений элемента в кортеж
# Заданный кортеж
A = ("ab", "ac", "ab", "ab", "ca", "ad", "jklmn")

d1 = A.count("ab")   # d1 = 3
d2 = A.count("jprst") # d2 = 0
d3 = A.count("ca")   # d3 = 1

print("d1 = ", d1)
print("d2 = ", d2)
print("d3 = ", d3)
```

Результат работы программы

```
d1 = 3
d2 = 0
d3 = 1
```

⇑

Связанные темы

- Кортежи. Основные понятия. Свойства кортежей

⇑

PYTHON КОНКАТЕНАЦИЯ КОРТЕЖ МЕТОД ОПЕРАЦИЯ ПОВТОРЕНИЕ СРЕЗ