

Множества, кортежи и вложенные коллекции. Шпаргалка

Множество

Создание множества:

```
# Как создать множество
my_set = {1, 2, 3, 4, 5}
print(my_set)
>>> {1, 2, 3, 4, 5}

my_set = set([1, 2, 3, 4, 5])
print(my_set)
>>> {1, 2, 3, 4, 5}

my_set = set()
print(my_set)
>>> set() # Пустое множество

my_set = {"Python"}
print(my_set)
>>> {'Python'}

my_set = set(["Python"])
print(my_set)
print(my_set)
>>> {'Python'}

# Так строка разбивается на символы — элементы множества
my_set = set("Python")
print(my_set)
>>> {'n', 't', 'h', 'P', 'y', 'o'}

# В множестве хранятся только уникальные элементы
my_set = {1, 2, 3, 4, 5, 5, 5}
print(my_set)
>>> {1, 2, 3, 4, 5}

# Так создать множество нельзя
my_set = {}
type(my_set)
>>> dict # Создается словарь
```

```
# Ошибка при создании множества
my_set = set(1, 2, 3, 4, 5)
print(my_set)
>>> TypeError... # Ошибка
```

Как добавить элемент во множество:

```
my_set = {1, 2, 3}
my_set.add(4)

print(my_set)
>>> {1, 2, 3, 4}
```

Как удалить конкретный элемент из множества:

```
my_set = {1, 2, 3, 4}
my_set.remove(3)

print(my_set)
>>> {1, 2, 4}
```



Если элемент не будет найден, мы получим исключение `KeyError`.

Чтобы избежать ошибки, нужно воспользоваться методом `discard()`:

```
my_set = {1, 2, 3, 4}
my_set.discard(5)

print(my_set)
>>> {1, 2, 3, 4}
```

Как удалить первый элемент из множества:

```
my_set = {1, 2, 3, 4}
element = my_set.pop()

print(my_set)
print(element)

>>> {2, 3, 4}
1
```

Как очистить множество от элементов:

```
my_set = {1, 2, 3, 4}
my_set.clear()

print(my_set)
>>> set()
```

Как объединить два множества:

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
union_set = set1.union(set2)

print(union_set)
>>> {1, 2, 3, 4, 5}
```



Помните, что множество хранит только уникальные элементы! Если в обоих множествах будет одинаковый элемент, в объединенном множестве останется только один такой элемент.

Как создать новое множество, в котором будут только элементы, которые есть в двух множествах:

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
intersection_set = set1.intersection(set2)

print(intersection_set)
>>> {3}
```

Как создать новое множество, в котором будут элементы из первого множества, которых нет во втором



При вызове метода важно, к какому множеству он применяется.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
difference_set = set1.difference(set2)

print(difference_set)
>>> {1, 2}

difference_set2 = set2.difference(set1)

print(difference_set2)
>>> {4, 5}
```

Как запустить цикл по множеству:

```
my_set = {1, 2, 3, 4, 5}

for element in my_set:
    print(element)
```

Как преобразовать множество в список:

```
my_set = {1, 2, 3, 4, 5}
my_list = list(my_set)
```

Как проверить, есть ли элемент в множестве:

```
my_set = {1, 2, 3, 4, 5}

if 3 in my_set:
    print("Элемент 3 есть в множестве.")
```

Кортеж

Как создать пустой кортеж:

```
my_tuple = tuple()
```

Как создать кортеж с одним элементом:

```
single_element_tuple = (1,)
```

Как посчитать количество вхождений элемента в кортеже:

```
my_tuple = (1, 2, 2, 3, 2, 4)
count_of_2 = my_tuple.count(2)

print(count_of_2)
>>> 3
```

Как найти индекс элемента в кортеже:

```
my_tuple = (1, 2, 3, 4, 5)
index_of_3 = my_tuple.index(3)

print(index_of_3)
>>> 2
```

Как посчитать длину кортежа:

```
my_tuple = (1, 2, 3, 4, 5)
length = len(my_tuple)

print(length)
>>> 5
```

Как найти максимальный элемент кортежа:

```
my_tuple = (1, 2, 3, 4, 5)
maximum = max(my_tuple)

print(maximum)
>>> 5
```

Как найти минимальный элемент кортежа:

```
my_tuple = (1, 2, 3, 4, 5)
minimum = min(my_tuple)

print(minimum)
>>> 1
```

Как отсортировать кортеж:

```
my_tuple = (3, 1, 4, 1, 5, 9, 2)
sorted_list = sorted(my_tuple)

print(sorted_list)
>>> [1, 1, 2, 3, 4, 5, 9]
```



Обратите внимание, что после сортировки возвращается список.

Вложенные коллекции

Как получить элементы из списка списков:

```
print(nested_list[0])
>>> [1, 2, 3]

print(nested_list[1][2])
>>> 6
```

Как получить элементы из списка кортежа кортежей:

```
print(nested_tuple[0])
>>> (1, 2, 3)

print(nested_tuple[1][2])
>>> 6
```

Как получить элементы из словаря словарей:

```
print(nested_dict['first'])
>>> {'one': 1, 'two': 2}

print(nested_dict['second']['three'])
>>> 3
```

Как получить элементы из списка словарей:

```
print(list_of_dicts[0]['name'])
>>> Alice
```

Как отсортировать список словарей по определенному ключу:

```
students = [
    {"name": "Alice", "age": 25},
    {"name": "Bob", "age": 22},
    {"name": "Eve", "age": 28}
]

sorted_students = sorted(students, key=lambda student: student["age"])
print(sorted_students)

>>> [
    {'name': 'Bob', 'age': 22},
    {'name': 'Alice', 'age': 25},
    {'name': 'Eve', 'age': 28}
]
```