

Методы списков

Операция	Описание	Пример
<code>x in a</code>	Проверка, что <code>x</code> содержится в <code>a</code>	<code>5 in [2, 3, 5]</code>
<code>x not in a</code>	Проверка, что <code>x</code> не содержится в <code>a</code> То же, что и <code>not (x in a)</code>	<code>5 not in [2, 3, 6]</code>
<code>a + a2</code>	Конкатенация списков, то есть новый список, в котором сначала идут все элементы <code>a</code> , а затем все элементы <code>a2</code>	<code>[2, 4] + [5, 3] == [2, 4, 5, 3]</code>
<code>a * k</code>	Список <code>a</code> , повторенный <code>k</code> раз	<code>[2, 3] * 3 == [2, 3, 2, 3, 2, 3]</code>
<code>a[n]</code>	<code>n</code> -й элемент списка, отрицательные <code>n</code> — для отсчета с конца	<code>[2, 3, 7][0] == 2</code> <code>[2, 3, 7][-1] == 7</code>
<code>a[start:stop:step]</code>	Срез списка	<code>[2, 3, 7][:2] == [2, 3]</code>
<code>len(a)</code>	Длина списка	<code>len([2, 3, 7]) == 3</code>
<code>max(a)</code>	Максимальный элемент списка	<code>max([2, 3, 7]) == 7</code>
<code>min(a)</code>	Минимальный элемент списка	<code>min([2, 3, 7]) == 2</code>
<code>sum(a)</code>	Сумма элементов списка	<code>sum([2, 3, 7]) == 12</code>
<code>a.index(x)</code>	Индекс первого вхождения <code>x</code> в <code>a</code> (вызовет ошибку, если <code>x not in a</code> , то есть если <code>x</code> отсутствует в <code>a</code>)	<code>[2, 3, 7].index(7) == 2</code>
<code>a.count(x)</code>	Количество вхождений <code>x</code> в <code>a</code>	<code>[2, 7, 3, 7].count(7) == 2</code>
<code>a.append(x)</code>	Добавить <code>x</code> в конец <code>a</code>	<code>a = [2, 3, 7]</code> <code>a.append(8)</code> <code>a == [2, 3, 7, 8]</code>
<code>a.extend(a2)</code>	Добавить элементы коллекции <code>a2</code> в конец <code>a</code>	<code>a = [2, 3, 7]</code> <code>a.extend([8, 4, 5])</code> <code>a == [2, 3, 7, 8, 4, 5]</code>
<code>del a[n]</code>	Удалить <code>n</code> -й элемент списка	<code>a = [2, 3, 7]</code> <code>del a[1]</code> <code>a == [2, 7]</code>
<code>del a[start:stop:step]</code>	Удалить из <code>a</code> все элементы, попавшие в срез	<code>a = [2, 3, 7]</code> <code>del a[:2]</code> <code>a == [7]</code>
<code>a.clear()</code>	Удалить из <code>a</code> все элементы (то же, что <code>del a[:]</code>)	<code>a.clear()</code>
<code>a.copy()</code>	Копия <code>a</code> (то же, что и полный срез <code>a[:]</code>)	<code>b = a.copy()</code>
<code>a += a2</code> <code>a *= k</code>	Заменить содержимое списка на <code>a + a2</code> и <code>a * k</code> соответственно	
<code>a.insert(n, x)</code>	Вставить <code>x</code> в <code>a</code> на позицию <code>n</code> , подвинув последующую часть дальше	<code>a = [2, 3, 7]</code> <code>a.insert(0, 8)</code> <code>a == [8, 2, 3, 7]</code>
<code>a.pop(n)</code>	Получить <code>n</code> -й элемент списка и одновременно удалить его из списка. Вызов метода без аргументов	<code>a = [2, 3, 7]</code> <code>a.pop(1) == 3</code> <code>a == [2, 7]</code>

	равносилен удалению последнего элемента: <code>a.pop() == a.pop(-1)</code>	
<code>a.remove(x)</code>	Удалить первое вхождение <code>x</code> в <code>a</code> , в случае <code>x not in a</code> — ошибка	<code>a = [2, 3, 7]</code> <code>a.remove(3)</code> <code>a == [2, 7]</code>
<code>a.reverse()</code>	Изменить порядок элементов в <code>a</code> на обратный (перевернуть список)	<code>a = [2, 3, 7]</code> <code>a.reverse()</code> <code>a == [7, 3, 2]</code>
<code>a.sort()</code>	Отсортировать список по возрастанию	<code>a = [3, 2, 7]</code> <code>a.sort()</code> <code>a == [2, 3, 7]</code>
<code>a.sort(reverse=True)</code>	Отсортировать список по убыванию	<code>a = [3, 2, 7]</code> <code>a.sort(reverse = True)</code> <code>a == [7, 3, 2]</code>
<code>bool(a)</code>	Один из способов проверить список на пустоту (возвращает <code>True</code> , если список непустой, и <code>False</code> в противном случае)	

Методы строк

Операция	Описание	Пример
<code>s2 in s</code>	Проверка, что подстрока <code>s2</code> содержится в <code>s</code>	<code>'m' in 'team'</code>
<code>s2 not in s</code>	Проверка, что подстрока <code>s2</code> не содержится в <code>s</code> то же, что <code>not (s2 in s)</code>	<code>'I' not in 'team'</code>
<code>s + s2</code>	Конкатенация (склейка) строк, то есть строка, в которой сначала идут все символы из <code>s</code> , а затем все символы из <code>s2</code>	<code>'tea' + 'm' == 'team'</code>
<code>s * k</code>	Строка <code>s</code> , повторенная <code>k</code> раз	<code>'ha' * 3 == 'hahaha'</code>
<code>s[n]</code>	<u>n</u> -й элемент строки, отрицательные <code>n</code> — для отсчета с конца	<code>'team'[2] == 'a'</code> <code>'team'[-1] == 'm'</code>
<code>s[start:stop:step]</code>	Срез строки	<code>'mama'[:2] == 'ma'</code>
<code>len(s)</code>	Длина строки	<code>len('abracadabra') == 11</code>
<code>s.find(s2)</code> <code>s.rfind(s2)</code>	Индекс начала первого или последнего вхождения подстроки <code>s2</code> в <code>s</code> (вернет <code>-1</code> , если <code>s2 not in s</code>)	<code>s = 'abracadabra'</code> <code>s.find('ab') == 0</code> <code>s.rfind('ab') == 7</code> <code>s.find('x') == -1</code>
<code>s.count(s2)</code>	Количество неперекрывающихся вхождений <code>s2</code> в <code>s</code>	<code>'abracadabra'.count('a') == 5</code>
<code>s.startswith(s2)</code> <code>s.endswith(s2)</code>	Проверка, что <code>s</code> начинается с <code>s2</code> или оканчивается на <code>s2</code>	<code>'abracadabra'.startswith('abra')</code> <code>'abracadabra'.endswith('bra')</code>

<code>s += s2</code> <code>s *= k</code>	Замениť содержимое строки на <code>s + s2</code> и <code>s * k</code> соответственно	
<code>s.isdigit()</code> <code>s.isalpha()</code> <code>s.isalnum()</code>	Проверка, что в строке <code>s</code> все символы — цифры, буквы (включая кириллические), цифры или буквы соответственно	<code>'100'.isdigit()</code> <code>'abc'.isalpha()</code> <code>'E315'.isalnum()</code>
<code>s.isdecimal()</code>	Возвращает <u>True</u> , если все символы в строке десятичные и есть хотя бы один символ, в противном случае <u>False</u>	<code>'123'.isdecimal()</code>
<code>s.islower()</code> <code>s.isupper()</code>	Проверка, что в строке <code>s</code> не встречаются большие буквы, маленькие буквы. Обратите внимание, что для обеих этих функций знаки препинания и цифры дают <u>True</u>	<code>'hello!'.islower()</code> <code>'123PYTHON'.isupper()</code>
<code>s.lower()</code> <code>s.upper()</code>	Строка <code>s</code> , в которой все буквы (включая кириллические) приведены к верхнему или нижнему регистру, т. е. заменены на строчные (маленькие) или заглавные (большие)	<code>'Привет!'.lower() == 'привет!'</code> <code>'Привет!'.upper() == 'ПРИВЕТ!'</code>
<code>s.capitalize()</code>	Возвращает копию строки <code>s</code> , в которой первая буква — заглавная (прописная), остальные строчные	<code>'привет'.capitalize() == 'Привет'</code>
<code>s.lstrip()</code> <code>s.rstrip()</code> <code>s.strip()</code>	Строка <code>s</code> , у которой удалены символы пустого пространства (пробелы, табуляции) в начале, в конце или с обеих сторон	<code>' Привет! '.strip() == 'Привет!'</code>
<code>s.ljust(k, c)</code> <code>s.rjust(k, c)</code>	Добавляет справа или слева нужное количество символов <code>c</code> , чтобы длина <code>s</code> достигла <code>k</code>	<code>'Привет'.ljust(8, '!') == 'Привет!!'</code>
<code>s.join(a)</code>	Склеивает строки из списка <code>a</code> через символ <code>s</code>	<code>'+'.join(['Вася', 'Маша']) == 'Вася+Маша'</code>
<code>s.split(s2)</code>	Список всех слов строки <code>s</code> (подстрок, разделенных строками <code>s2</code>)	<code>'Раз два три!'.split('а') == ['Р', 'з дв', ' три!']</code>

<code>s.replace(s2, s3)</code>	Строка <code>s</code> , в которой все неперекрывающиеся вхождения <code>s2</code> заменены на <code>s3</code> Есть необязательный третий параметр, с помощью которого можно указать, сколько раз производить замену	<code>'Раз два три!'.replace('а', 'я') == 'Ряз два три!'</code> <code>'Раз два три!'.replace('а', 'я', 1) == 'Ряз два три!'</code>
<code>list(s)</code>	Список символов из строки <code>s</code>	<code>list('Привет') == ['П', 'р', 'и', 'в', 'е', 'т']</code>
<code>bool(s)</code>	Проверка, что строка не пустая (возвращает <code>True</code> , если не пустая, и <code>False</code> в противном случае)	
<code>int(s)</code> <code>float(s)</code>	Если в строке <code>s</code> записано целое (дробное) число, получить это число, иначе — ошибка	<code>int('25') == 25</code>
<code>str(x)</code>	Представить любой объект <code>x</code> в виде строки	<code>str(25) == '25'</code>

Важно!

Обратите внимание: **никакие** методы строк, включая `s.replace(...)`, не изменяют саму строку `s`. Все они лишь **возвращают измененную строку**, в отличие от большинства методов списков.

`a.sort()`, например, ничего не возвращает, а изменяет сам список `a`.