**Срезы списков**

**Срезы**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| sp=[ | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 | ] |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |  |

1. **sp[a:b]** из списка sp берутся элементы с индексами от a до b, не включая b.

Длина списка b-a. Индексацию можно использовать как положительную, так и отрицательную.

sp[3:7] # [40, 50, 60, 70]

sp[5:1] пустой список

sp[1:-1] список без первого и последнего элемента

В срезе можно опускать первую или последнюю часть

sp[:4] # [10, 20, 30, 40]

sp[:n] первые n элементов

sp[5:] # [60, 70, 80, 90, 100, 110, 120] с 5 элемента и до конца

sp[-5:] последние 5 элементов

sp[:] копия списка

**2. sp[a:b:d]** из списка берутся элементы, начиная с a до b (не включая b) с шагом d

Длина списка (b-a)//d. Могут отсутствовать некоторые части.

sp[3:9:2] # [40, 60, 80]

sp[3:10:2] # [40, 60, 80, 100]

sp[::2] элементы на четных местах

Если d<0, то срез разворачивается

sp[9:2:-2] # [100, 80, 60, 40]

sp[9:2:-1] # [100, 90, 80, 70, 60, 50, 40]

sp[::-1] разворот списка в обратном порядке

Срезы можно комбинировать

Отбросить крайние элементы и развернуть sp[-2:0:-1] или sp[1:-1][::-1]

Срезы можно брать и у строк. Срезы тоже будут являться копией.

## Операции и методы списков в Python

|  |  |
| --- | --- |
| x **in** a | проверка, что x содержится в a5 **in** [2, 3, 5] |
| x **not in** a | то же, что not (x in a)5 **not in** [2, 3, 7] |
| a + a2 | конкатенация списков, то есть список, в котором сначала идут все элементы a, а затем все элементы a2[2, 4] + [5, 3] == [2, 4, 5, 3] |
| a \* k | список a, повторенный k раз[2, 3] \* 3 == [2, 3, 2, 3, 2, 3] |
| a[start:stop:step] | срез[2, 3, 7][:2] == [2, 3] |
| len(a) | длина спискаlen([2, 3, 7]) == 3 |
| max(a) | максимальный элементmax([2, 3, 7]) == 7 |
| min(a) | минимальный элементmin([2, 3, 7]) == 2 |
| sum(a) | сумма чисел в спискеsum([2, 3, 7]) == 12 |
| a.index(x) | индекс первого вхождения x в a (вызовет ошибку, если х отсутствует в а)[2, 3, 7].index(7) == 2 |
| a.count(x) | количество вхождений x в a[2, 3, 7].count(7) == 1 |
| a.append(x) | добавить x в конец aa = [2, 3, 7]a.append(8)a == [2, 3, 7, 8] |
| del a[n] | удалить n-й элемент спискаa = [2, 3, 7]del a[1]a == [2, 7] |
| del a[start:stop:step]  | удалить из a все элементы, попавшие в срезa = [2, 3, 7]del a[:2]a == [7] |
| a += a2a \*= k | заменить содержимое списка на a+a2 и a\*k, соответственно |
| a.insert(n, x) | вставить x в a на индекс (позицию) n, подвинув последующую часть дальшеa = [2, 3, 7]a.insert(0, 8)a == [8, 2, 3, 7] |
| a.pop() | получить последний элемент списка и одновременно удалить его из спискаa = [2, 3, 7]a.pop() == 7a == [2, 3] |
| a.pop(n) | получить n-й элемент списка и одновременно удалить его из спискаa = [2, 3, 7]a.pop(1) == 3a == [2, 7] |
| a.reverse() | изменить порядок элементов в a на обратный (перевернуть список)a = [2, 3, 7]a.reverse()a == [7, 3, 2] |
| a.sort() | отсортировать элементы по возрастаниюa = [3, 2, 7]a.sort()a == [2, 3, 7] |
| a.sort(reverse = True) | отсортировать элементы по убываниюa = [3, 2, 7]a.sort(reverse = True)a == [7, 3, 2] |