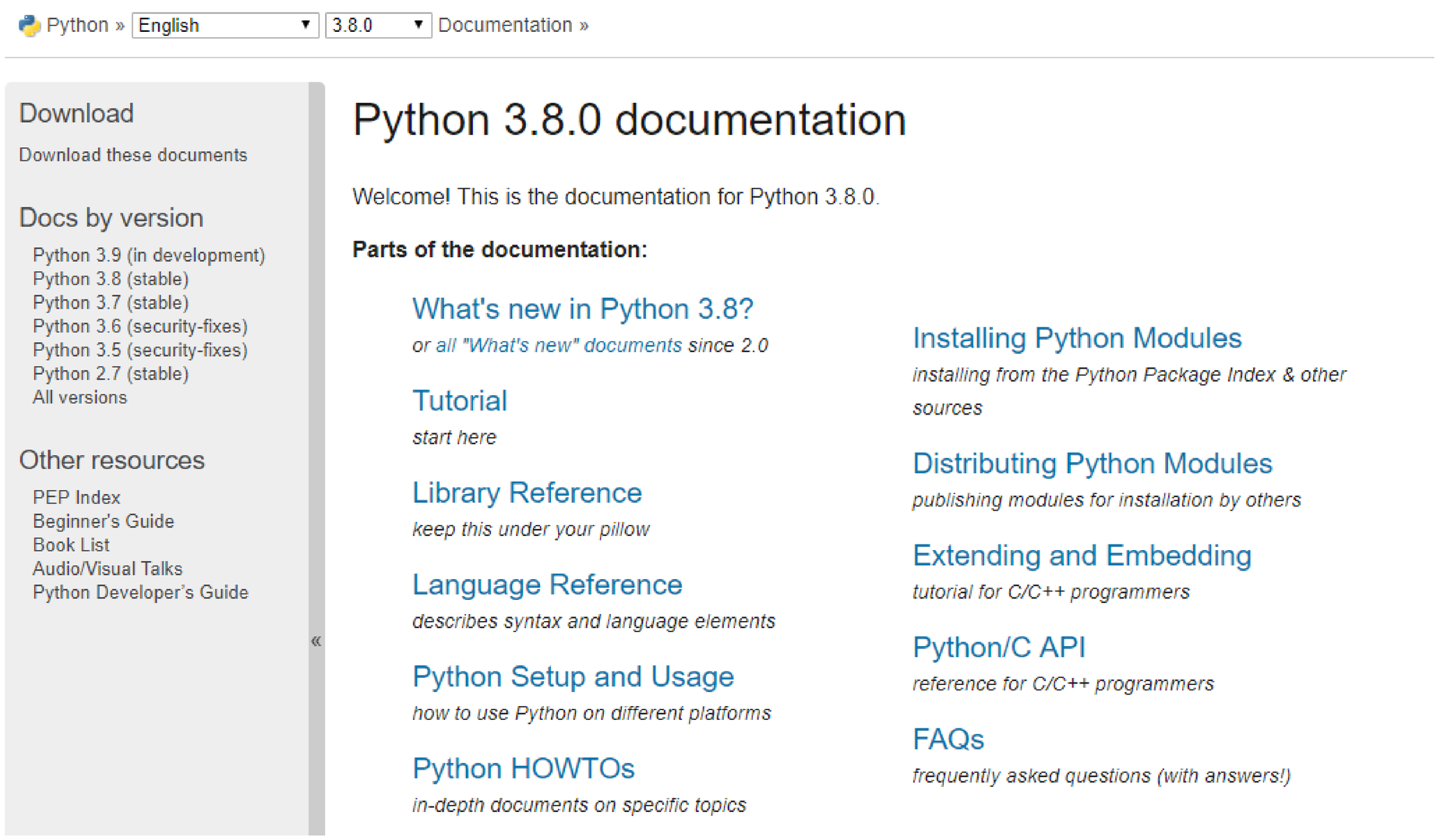

Яндекс

Яндекс Лицей

Методы списков и строк

Всё есть в документации



Методы списков

Операция	Описание	Пример
x in a	Проверка, что x содержится в a	5 in [2, 3, 5]
x not in a	Проверка, что x не содержится в a То же, что и not (x in a)	5 not in [2, 3, 5]
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[n]	n-й элемент списка, отрицательные n — для отсчёта с конца	[2, 3, 7][0] == 2 [2, 3, 7][-1] == 7
a[start:stop:step]	срез	[2, 3, 7][:2] == [2, 3]
len(a)	Длина списка	len([2, 3, 7]) == 3
max(a)	Максимальный элемент списка	max([2, 3, 7]) == 7
min(a)	Минимальный элемент списка	min([2, 3, 7]) == 2
sum(a)	Сумма элементов списка	sum([2, 3, 7]) == 12

Выделение **жирным** символизирует, что эти методы изменяют сам список a

a.index(x)	Индекс первого вхождения x в a (вызовет ошибку, если x not in a, то есть если x отсутствует в a)	[2, 3, 7].index(7) == 2
a.count(x)	Количество вхождений x в a	[2, 7, 3, 7].count(7) == 2
a.append(x)	Добавить x в конец a	a = [2, 3, 7] a.append(8) a == [2, 3, 7, 8]
a. extend(a2)	Добавить элементы коллекции a2 в конец a	a = [2, 3, 7] a.extend([8, 4, 5]) a == [2, 3, 7, 8, 4, 5]
del a[n]	Удалить n-й элемент списка	a = [2, 3, 7] del a[1] a == [2, 7]
del a[start:stop:step]	Удалить из a все элементы, попавшие в срез	a = [2, 3, 7] del a[:2] a == [7]
a.clear()	Удалить из a все элементы (то же, что del a[:])	a.clear()
a.copy()	Копия a (то же, что a[:])	b = a.copy()

a += a2 **a *= k**	Заменить содержимое списка на a + a2 и a * k соответственно	
a.insert(n, x)	Вставить x в a на позицию n, подвинув последующую часть дальше	a = [2, 3, 7] a.insert(0, 8) a == [8, 2, 3, 7]
a.pop(n)	Получить n-й элемент списка и одновременно удалить его из списка a.pop() == a.pop(-1)	a = [2, 3, 7] a.pop(1) == 3 a == [2, 7]
a.remove(x)	Удалить первое вхождение x в a, в случае x not in a — ошибка	a = [2, 3, 7] a.remove(3) a == [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 список по возрастанию	a = [3, 2, 7] a.sort() a == [2, 3, 7]
a.sort(reverse=True)	Отсортировать список по убыванию	a = [3, 2, 7] a.sort(reverse = True) a == [7, 3, 2]
bool(a)	Один из способов проверить список на пустоту	

Методы строк

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

Методы строк

s.startswith(s2) s.endswith(s2)	Проверка, что s начинается с s2 или оканчивается на s2	'abracadabra'.startswith('abra')
s += s2 s *= k	Заменить содержимое строки s на s + s2 и s * k соответственно	
s.isdigit() s.isalpha() s.isalnum()	Проверка, что в строке s все символы — цифры, буквы (включая кириллические), цифры или буквы, соответственно	'100'.isdigit() 'abc'.isalpha() 'E315'.isalnum()
s.islower() s.isupper()	Проверка, что в строке s не встречаются большие буквы, маленькие буквы. Обратите внимание, что для обеих этих функций знаки препинания и цифры дают True	'hello!'.islower() '123PYTHON'.isupper()
s.lower() s.upper()	Строка s, в которой все буквы (включая кириллические) приведены к верхнему или нижнему регистру, т.е. заменены на строчные (маленькие) или заглавные (большие)	'Привет!'.lower() == 'привет!' ''Привет!'.upper() == 'ПРИВЕТ!'
s.capitalize()	Строка s, в которой первая буква — заглавная	'привет'.capitalize() == 'Привет'

Методы строк

s.lstrip() s.rstrip() s.strip()	Строка s, у которой удалены символы пустого пространства (пробелы, табуляции) в начале, в конце или с обеих сторон	' Привет! '.strip() == 'Привет!'
s.ljust(k, c) s.rjust(k, c)	Добавляет справа или слева нужное количество символов c, чтобы длина s достигла k	'Привет'.ljust(8, '!') == 'Привет!!'
s.join(a)	a — список строк, тогда s.join(a) — эти строки, «склеенные» через s	'+'.join(['Вася', 'Маша']) == 'Вася+Маша'
s.split(s2)	Список слов строки s (подстрок, разделённых строками s2)	'Раз два три!'.split('а') == ['Р', 'з дв', ' три!']
s.replace(s2, s3)	Строка s, в которой все неперекрывающиеся вхождения s2заменены на s3	'Раз два три!'.replace('а', 'я') == 'Ряз двя три!'

Методы строк

list(s)	Список символов из строки строки s	list('Привет') == ['П', 'р', 'и', 'в', 'е', 'т']
bool(s)	Проверка, что строка не пустая	
int(s) float(s)	Если в строке s записано целое (дробное) число, получить это число, иначе ошибка	int('25') == 25
str(x)	Получить какое-то представление в виде строки любого объекта x	str(25) == '25'

Функции dir и help

Функция dir возвращает список методов любого объекта:

```
print(dir([]))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__',
'__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__',
'__getitem__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__',
'__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mul__',
'__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__reversed__',
'__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__',
'__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index',
'insert', 'pop', 'remove', 'reverse', 'sort']
```

Функции dir и help

Функция help выводит справку по любому объекту или методу:

```
help([].insert)

Help on built-in function insert:

insert(...) method of builtins.list instance
    L.insert(index, object) -- insert object before index
```

f-строки

f-строки, которые буквально означают formatted string, задаются с помощью литерала f перед кавычками:

```
name = "Аркадий"
age = 14
print(f"Меня зовут {name} Мне {age} лет.")
```

```
Меня зовут Аркадий. Мне 14 лет.
```

f-строки

В качестве подставляемого значения внутри фигурных скобок может быть:

```
f"Меня зовут {name}"                    # имя переменной

f"Третий месяц в году - {month[2]}"     # элемент списка или словаря

f"Имя: {name.upper()}"                  # методы объектов
```

f-строки

f-строки позволяют выполнять базовые арифметические операции:

```
f"({x} + {y}) / 2 = {(x + y) / 2}"
```

И даже вызывать функции:

```
f"13 / 7 = {round(13/7)}"
```

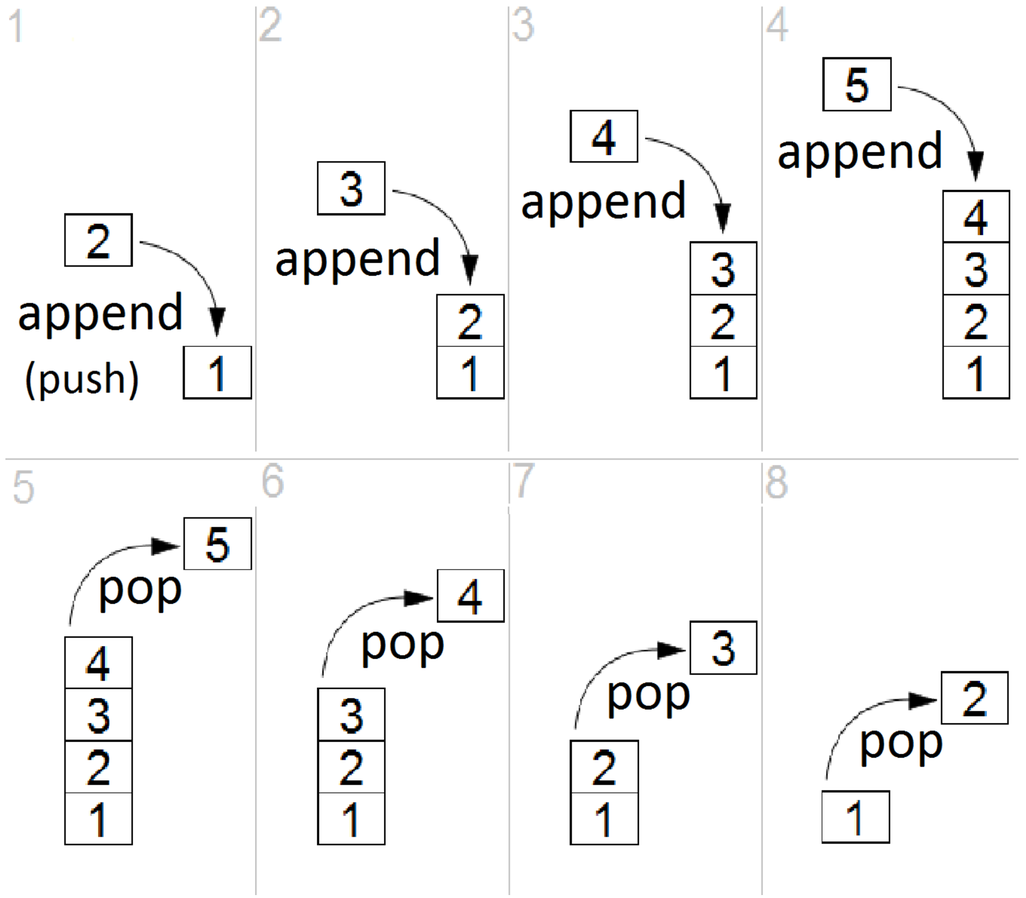
Цепочки вызовов методов

```
s                                                   ' Зелёный клён '
s.strip()                                           'Зелёный клён'
s.strip().lower()                                   'зелёный клён'
s.strip().lower().replace('ё','e')                  'зеленый клен'
```

```
'мало Средне МНОГО'                                 'мало Средне МНОГО‘
'мало Средне МНОГО'.lower()                         'мало средне много'
'мало Средне МНОГО'.lower().split()                 ['мало', 'средне', 'много']
'мало Средне МНОГО'.lower().split().index('средне') 1
```

Стек

stack – «стопка»: что добавлено последним, убирается первым



Стек

```
stack = []
for i in range(5):
    print('Какую футболку вы кладёте сверху
                                        стопки?')
    stack.append(input())
while stack: # пока стопка не закончилась
    print('Сегодня вы надеваете футболку',
                                      stack.pop())
```

```
Какую футболку вы кладёте сверху стопки?
белую
Какую футболку вы кладёте сверху стопки?
красную
Какую футболку вы кладёте сверху стопки?
синюю
Какую футболку вы кладёте сверху стопки?
зеленую
Какую футболку вы кладёте сверху стопки?
веселую
Сегодня вы надеваете футболку веселую
Сегодня вы надеваете футболку зеленую
Сегодня вы надеваете футболку синюю
Сегодня вы надеваете футболку красную
Сегодня вы надеваете футболку белую
```

Решение задачи «Найди кота – 6»

```
n = int(input())
for i in range(n):
    s = input()
    position = s.find('кот')
        if position != -1:
            print(i + 1, position + 1)
```

Решение задачи «Слово для Гиннесса»

```
words = input().split()
print(max(len(word) for word in words))
```

Решение задач «От и до»

```
a = [int(x) for x in input().split()]
m, k = [int(x) for x in input().split()]
print(sum(a[m:k + 1]))
```

Решение задач «От и до»

```
a = [int(x) for x in input().split()]
m, k = [int(x) for x in input().split()]
print(sum(a[m:k+1]))
```

```
a = [int(x) for x in input().split()]
m, k = [int(x) for x in input().split()]
print(sum(x ** 2 for x in a[m:k + 1]))
```

Решение задачи «Комментарии в программе»

```
n = int(input()[1:])
for _ in range(n):
    s = input()
    if '#' in s:
        s = s[:s.find('#')]
    s = s.rstrip()
    print(s)
```


Яндекс