

PYTHON

Структуры данных. Функции. БД.

Классы и объекты

```
>>> a = 5
```

```
>>> type(a)
```

```
<class 'int'>
```

```
>>> a = 'Hell!o'
```

```
>>> print(a)
```

```
Hell!o
```

```
>>> print(a.replace('!o','o!')) #метод replace()
```

```
Hello!
```

```
>>> print(a)
```

```
Hell!o
```

Строки (STRING): создание

```
>>> a = 'String1'
```

```
>>> a = "String2"
```

```
>>> a = "Some
```

```
... long text"
```

```
>>> a = ""Some long
```

```
... text 2""
```

```
>>> print(a)
```

```
Some long
```

```
text 2
```

Строки: операции и сравнение

```
>>> a = 'x'+'o'
```

```
>>> print(a)
```

```
xo
```

```
>>> a = a * 3
```

```
>>> print(a)
```

```
xoxoxo
```

```
>>> 'qwe' == "qwe"
```

```
True
```

```
>>> 'abc' < 'abd'
```

```
True
```

```
>>> 'abc' > 'abd'
```

```
False
```

```
>>> 'abc' < 'ab'
```

```
False
```

```
>>> 'abc' > 'ab'
```

```
True
```

Строки: разбор на символы

```
a = 'Python'
```

```
a[0] = 'P'
```

```
a[-1] = 'n'
```

```
i = 2
```

```
a[1] = 'y'
```

```
a[-2] = 'o'
```

```
>>> print(a[i])
```

```
a[2] = 't'
```

```
a[-3] = 'h'
```

```
t
```

```
a[3] = 'h'
```

```
a[-4] = 't'
```

```
a[4] = 'o'
```

```
a[-5] = 'y'
```

```
>>> a[i] = 'H' #Ошибка!
```

```
a[5] = 'n'
```

```
a[-6] = 'P'
```

Строки: циклы

```
a = 'Python'  
for i in range(6):  
    print(a[i])
```

```
a = 'Python'  
for i in range(len(a)):  
    print(a[i])
```

```
a = 'Python'  
for c in a:  
    print(c)
```

Вывод у всех трех:

P
y
t
h
o
n

Строки: задача

Посчитаем сколько раз встречается символ **о** в введенной строке.

Решение:

```
a = input('Введите строку: ')
n = 0
for c in a:
    if c == 'о':
        n+=1
print(n)
```

Строки: еще одно решение задачи

```
a = input('Введите строку: ')\nprint(a.count('o'))
```


Строки: методы

```
a = 'Python'
```

```
b = 'th'
```

```
print(a.upper())
```

PYTHON

```
print(a.lower())
```

python

```
print(a.count(b))
```

1

```
print(a.find(b))
```

2

```
print(a.replace(b, 'TH'))
```

PyTHon

Строки: последовательность методов

```
a = 'Guido van RosSum'
```

```
b = 's'
```

```
print(a.upper().count(b.upper()))
```

Строки: slicing

```
>>> a = 'GuidoVanRossum'
```

```
>>> len(a)      14
```

```
>>> a[1]         'u'
```

```
>>> a[1:5]       'uido'
```

```
>>> a[:5]        'Guido'
```

```
>>> a[5:]        'VanRossum'
```

```
>>> a[-5:]       'ossum'
```

```
>>> a[1:-1]      'uidoVanRossu'
```

```
>>> a[1:-1:2]    'udVnos'
```

```
>>> a[::-1]      'mussoRnaVodiuG'
```

Списки (LIST)

```
>>> friends = ['Sasha', 'Dasha', 'Pasha', 'Masha']
```

```
>>> a = []
```

```
>>> friends[0]  'Sasha'      >>> friends[-1] 'Masha'
```

```
>>> friends[1]  'Dasha'      >>> friends[-2] 'Pasha'
```

```
>>> friends[2]  'Pasha'      >>> friends[-3] 'Dasha'
```

```
>>> friends[3]  'Masha'      >>> friends[-4] 'Sasha'
```

```
>>> friends[1:3]    ['Dasha', 'Pasha']
```

```
>>> friends[::-1]    ['Masha', 'Pasha', 'Dasha', 'Sasha']
```

Списки: перебор элементов и проверка вхождения

```
friends = ['Sasha', 'Dasha', 'Pasha', 'Masha']  
  
for item in friends:  
    print('Hello, ', item)
```

```
Hello, Sasha  
Hello, Dasha  
Hello, Pasha  
Hello, Masha
```

```
if 'Pasha' in friends:  
    print('Pasha in list.')
```



```
if 'Gosha' not in friends:  
    print('Gosha is not in list')
```

```
>>> friends.index('Pasha')      2  
>>> friends.index('Gosha')
```



```
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
ValueError: 'Gosha' is not in list
```

Списки: операции

```
>>> a = [1, 2, 3]
```

```
>>> friends+a
```

```
['Sasha', 'Dasha', 'Pasha', 'Masha', 1, 2, 3]
```

```
>>> a * 3
```

```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

Списки: изменение элемента

```
>>> friends = ['Sasha','Dasha','Pasha','Masha']
```

```
>>> friends[1] = 'Gosha'
```

```
>>> print(friends)
```

```
['Sasha', 'Gosha', 'Pasha', 'Masha']
```

Списки: добавление элемента

```
>>> friends += ['Dasha']
```

```
>>> print(friends)
```

```
['Sasha', 'Gosha', 'Pasha', 'Masha', 'Dasha']
```

```
>>> friends.append('Masha')
```

```
>>> print(friends)
```

```
['Sasha', 'Gosha', 'Pasha', 'Masha', 'Dasha', 'Masha']
```


Списки: вставка элемента

```
>>> friends = ['Sasha', 'Dasha', 'Pasha', 'Masha']
```

```
>>> friends.insert(2, 'Misha')
```

```
>>> print(friends)
```

```
['Sasha', 'Dasha', 'Misha', 'Pasha', 'Masha']
```

Списки: удаление

```
>>> print(friends)
```

```
['Sasha', 'Misha', 'Dasha', 'Pasha', 'Masha', 'Dasha']
```

```
>>> friends.remove('Dasha')
```

```
>>> print(friends)
```

```
['Sasha', 'Misha', 'Pasha', 'Masha', 'Dasha']
```

```
>>> print(friends)
```

```
['Sasha', 'Misha', 'Pasha', 'Masha', 'Dasha']
```

```
>>> del friends[1]
```

```
>>> print(friends)
```

```
['Sasha', 'Pasha', 'Masha', 'Dasha']
```

Списки: сортировка

```
>>> print(friends)
['Sasha', 'Pasha', 'Masha', 'Dasha']
>>> sorted(friends)
['Dasha', 'Masha', 'Pasha', 'Sasha']
>>> print(friends)
['Sasha', 'Pasha', 'Masha', 'Dasha']
```

```
>>> print(friends)
['Sasha', 'Pasha', 'Masha', 'Dasha']
>>> friends.sort()
>>> print(friends)
['Dasha', 'Masha', 'Pasha', 'Sasha']
```

Списки: обратный порядок

```
>>> print(friends)
['Sasha', 'Dasha', 'Pasha', 'Masha']
>>> a = list(reversed(friends))
>>> a
['Masha', 'Pasha', 'Dasha', 'Sasha']
```

```
>>> print(friends)
['Sasha', 'Dasha', 'Pasha', 'Masha']
>>> friends.reverse()
>>> print(friends)
['Masha', 'Pasha', 'Dasha', 'Sasha']
```

Списки: запись в другую переменную

```
>>> a = [3, 5, 7]
```

```
>>> b = a
```

```
>>> b    —    [3, 5, 7]
```

```
>>> a[1] = 9
```

```
>>> b    —    [3, 9, 7]
```

```
>>> b[0] = 11
```

```
>>> a    —    [11, 9, 7]
```

List comprehension и Generator expression

```
>>> a = [0 for i in range(3)]      [0, 0, 0]
```

```
>>> a = [i for i in range(3)]      [0, 1, 2]
```

```
>>> a = [int(i) for i in input().split()]
```

7 8 9 – ввели через пробел, получили:

```
[7, 8, 9]
```

```
>>> a = (i for i in range(3))
```

```
>>> a
```

```
<generator object <genexpr> at  
0x0000022595950350>
```

```
>>> for i in a:
```

```
...         print(i, end=' ')
```

```
0 1 2
```

Двумерные списки

```
>>> a = [[1,2,3],[4,5,6],[7,8,9]]
```

```
>>> a[1]
```

```
[4, 5, 6]
```

```
>>> a[1][2]
```

```
6
```

```
>>> for i in a:
```

```
...     print('\t'.join([str(j) for j in i]))
```

```
1      2      3
```

```
4      5      6
```

```
7      8      9
```

Двумерные списки и list comprehension

```
>>> a = [[i*3+j+1 for j in range(3)] for i in range(3)]
```

```
>>> for i in a:
```

```
...     print(i)
```

```
[1, 2, 3]
```

```
[4, 5, 6]
```

```
[7, 8, 9]
```


Множество (SET): создание и методы

```
>>> a = {'milk','butter','bread','cucumber'}
```

```
>>> a = {}
```

```
>>> a = set()
```

```
>>> a = {'milk','butter','bread','cucumber'}
```

```
>>> a.add('banana')           {'milk', 'banana', 'cucumber', 'butter', 'bread'}
```

```
>>> a.remove('cucumber')     {'milk', 'banana', 'butter', 'bread'}
```

```
>>> a.discard('tea')         {'milk', 'banana', 'butter', 'bread'}
```

```
>>> a.clear()
```

Множество: перебор элементов

```
>>> a = {'milk', 'butter', 'bread', 'cucumber'}
```

```
>>> for item in a:  
...     print(item)
```

milk

bread

cucumber

butter

Словари (DICTIONARY)

```
>>> a = {'sql': 'RDBMS', 'mongo': 'DO', 'redis': 'KV'}
```

```
>>> a = dict()
```

```
>>> a['sql']  
'RDBMS'
```

Словари: операции

```
>>> a = {'sql': 'RDBMS', 'mongo': 'DO', 'redis': 'KV'}
```

```
>>> a['oracle'] = 100
```

```
{'sql': 'RDBMS', 'mongo': 'DO', 'redis': 'KV', 'oracle': 100}
```

```
>>> a['oracle']          100
```

```
>>> a.get('oracle')     100
```

```
>>> del a[100]
```

```
>>> a['redis'] += 'RAM'
```

Словари: получение данных

```
>>> a = {'sql': 'RDBMS', 'mongo': 'DO', 'redis': 'KV', 'oracle': 100}
```

```
>>> for k in a: print(k, end=' ')
```

```
>>> for k in a.keys(): print(k, end=' ')
```

sql mongo redis oracle

```
>>> for v in a.values(): print(v, end=' ')
```

RDBMS DO KV 100

```
>>> for k, v in a.items():
```

```
    print(k, v, end=', ')
```

sql RDBMS, mongo DO, redis KV, oracle 100

Функции: объявление и вызов

объявление функции

```
def print_hi(name):
```

```
    print(f'Hi, {name}')
```

код программы

```
print_hi('PyCharm') # вызов функции
```

Функции: ВИДЫ

- 1) Не имеют параметров
- 2) Имеют произвольное число параметров
- 3) Не возвращают значение
- 4) Возвращают значение
- 5) Имеют параметры со значениями по умолчанию

```
1) def func_hi ():  
    print('Hi')  
2) def print(*values: object,  
    ...)  
3) def print_hi(name):  
    print(f'Hi, {name}')
```

```
4) def func_mp (a, b):  
    return a*b  
5) def func_mp (a, b=100):  
    return a*b
```

Функции: область видимости переменных

```
def func(a, b):
```

```
    x = 6
```

```
    return a*b
```

```
if __name__ == '__main__':
```

```
    a, b, x = 2, 3, 4
```

```
    print(func(a,b))
```

```
    print(x) → 4
```

```
def func(a):
```

```
    a.append(5)
```

```
if __name__ == '__main__':
```

```
    a = []
```

```
    func(a)
```

```
    print(a) → [5]
```

```
def func(a):
```

```
    print(a) → Glob
```

```
if __name__ == '__main__':
```

```
    a = 'Glob'
```

```
    func(a)
```


Python и Redis

```
C:\>python -m pip install redis
```

В коде программы объявляем объект класса Redis:

```
import redis
```

```
r = redis.Redis()  
r.mset({'sql': 'RDBMS', 'mongo': 'DO', 'redis': 'KVRAM', 'oracle': 300})  
print(r.mget('oracle'))
```

Python и Redis: что скрывается за классом Redis

Python

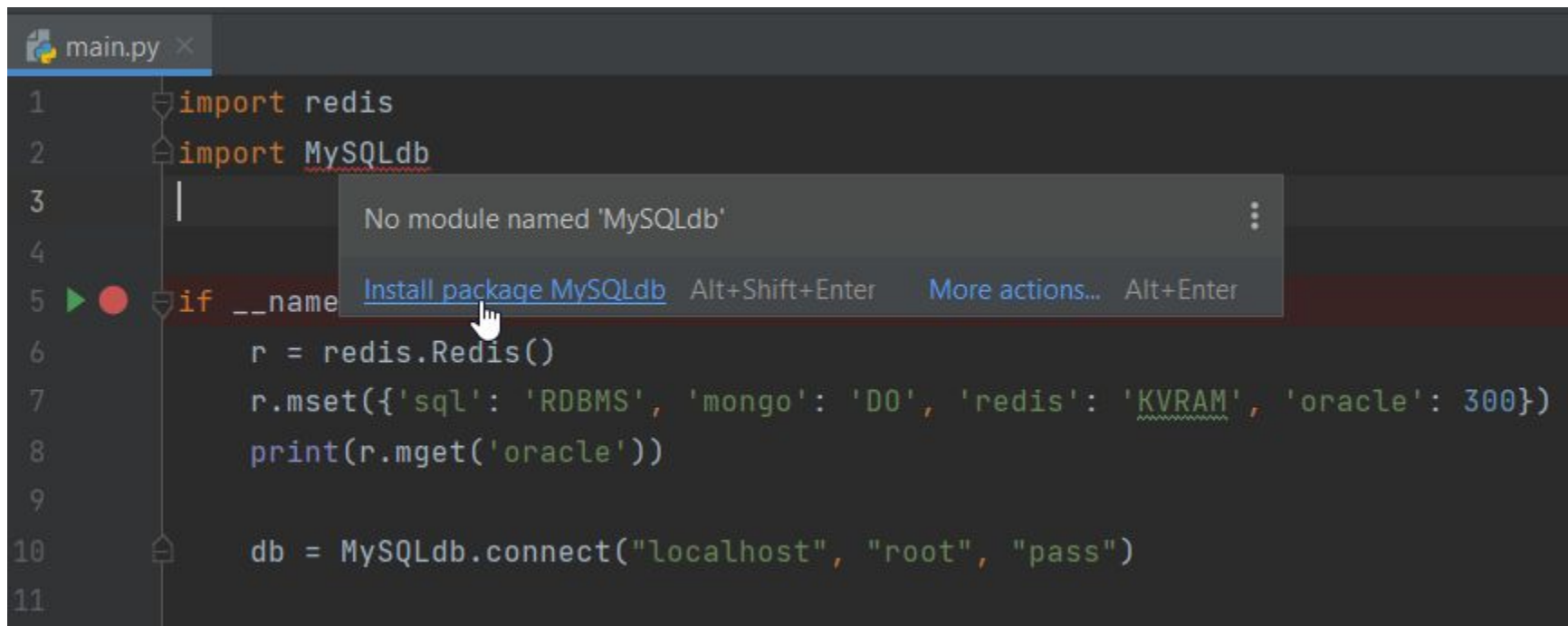
```
# From redis/client.py
class Redis(object):
    def __init__(self, host='localhost', port=6379,
                 db=0, password=None, socket_timeout=None,
                 # ...
```

```
C:\>docker exec -it redis-port redis-cli
```

```
127.0.0.1:6379> keys *
```

- 1) "oracle"
- 2) "mongo"
- 3) "sql"
- 4) "redis"

Установка библиотеки из IDE



The screenshot shows a code editor with a file named `main.py`. The code contains the following lines:

```
1 import redis
2 import MySQLdb
3
4
5 if __name__ == '__main__':
6     r = redis.Redis()
7     r.mset({'sql': 'RDBMS', 'mongo': 'D0', 'redis': 'KVRAM', 'oracle': 300})
8     print(r.mget('oracle'))
9
10 db = MySQLdb.connect("localhost", "root", "pass")
11
```

A red squiggly line under `MySQLdb` in line 2 indicates an error. A context menu is open over the error, displaying the message "No module named 'MySQLdb'". The menu includes the option [Install package MySQLdb](#) with the keyboard shortcut `Alt+Shift+Enter`, and a [More actions...](#) option with the shortcut `Alt+Enter`. A mouse cursor is pointing at the [Install package MySQLdb](#) option.