



Курс по основам  
программирования на Python

# Функции

**Функции** — это такие участки кода, которые изолированы от остальной программы и выполняются только тогда, когда вызываются.

- 1) Код функции должен размещаться в начале программы
- 2) Первая строка содержит идентификатор – имя функции
- 3) После идентификатора в круглых скобках идет список параметров, которые получает наша функция.
- 4) В конце строки двоеточие
- 5) Тело функции (с отступом)
- 6) Функция завершается инструкцией и возвращением значения (если нужно)

```
def max(a, b):  
    if a > b:  
        return a  
    else:  
        return b  
  
print(max(3, 5))  
print(max(5, 3))  
print(max(int(input()), int(input())))
```



# Функции

*0* Функция, которая принимает на вход три параметра и возвращает максимальное из них.

```
def max(a, b):  
    if a > b:  
        return a  
    else:  
        return b  
  
def max3(a, b, c):  
    return max(max(a, b), c)  
  
print(max3(3, 5, 4))
```

# Локальные и глобальные переменные.

- 0 Глобальные переменные – переменные объявленные вне функции, но доступные внутри функции.
- 0 Локальные переменные – переменные объявленные внутри функции (недоступные за пределами функции)



# Локальные и глобальные переменные.

0 «Защита» от случайных изменений

```
1 def f():  
2     a = 1  
3     print(a)  
4  
5 a = 0  
6 f()  
7 print(a)  
8 |
```

Выходные данные:

```
1 1  
2 0  
3 |
```

# Локальные и глобальные переменные.

```
def factorial(n):  
    res = 1  
    for i in range(1, n + 1):  
        res *= i  
    return res  
  
for i in range(1, 6):  
    print(i, '! = ', factorial(i), sep='')
```



# Локальные и глобальные переменные

```
def f():  
    global a  
    a = 1  
    print(a)  
  
a = 0  
f()  
print(a)
```

Выходные данные:

1	1
2	1

# Задача 1

Даны четыре действительных числа:  $x_1$ ,  $y_1$ ,  $x_2$ ,  $y_2$ . Напишите функцию `distance(x1, y1, x2, y2)`, вычисляющая расстояние между точкой  $(x_1, y_1)$  и  $(x_2, y_2)$ . Считайте четыре действительных числа и выведите результат работы этой функции.



# Решение

```
from math import sqrt
def distance(x1, y1, x2, y2):
    return sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)

x1 = float(input())
y1 = float(input())
x2 = float(input())
y2 = float(input())

print(distance(x1, y1, x2, y2))
```

## Задача 2

Дано действительное положительное число  $a$  и целое число  $n$  (допускается отрицательное).

Вычислите  $a$  в степени  $n$ . Решение оформите в виде функции `power(a, n)`.

Стандартной функцией возведения в степень пользоваться нельзя.



# Решение

```
def power(a, n):  
    res = 1  
    for i in range(abs(n)):  
        res *= a  
    if n < 0:  
        res = 1 / res  
    return res  
a = float(input())  
n = int(input())  
print(power(a, n))
```

# Рекурсия

- 0 Функция может вызывать не только другую функцию, но и саму себя!

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n - 1)  
  
print(factorial(5))
```



# Задача 1

Дана последовательность целых чисел, заканчивающаяся числом 0. Выведите эту последовательность в обратном порядке.

При решении этой задачи нельзя пользоваться массивами и прочими динамическими структурами данных. Рекурсия вам поможет.

# Решение

```
def reverse():  
    x = int(input())  
    if x != 0:  
        reverse()  
    print(x)  
  
reverse()
```