Python 3

middle

Буженко Александр Александрович

Функции

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

```python
def factorial(n):
    res = 1
    for i in range(1, n + 1):
        res *= i
    return res

print(factorial(3))
print(factorial(5))
```

У функции есть:

1. Имя(обязательно)
2. Возвращаемое значение (не обязательно)
3. Входящие параметры (не обязательно)

Имя - это идентификатор, то есть имя нашей функции по которому мы можем вызывать функцию

После идентификатора в круглых скобках идет список параметров, которые получает наша функция.

```
def sum(n1,n2,n3,n4,size):

        ...

        ...
```

Далее идет тело функции, оформленное в виде блока, то есть с отступом.

```
1  def factorial(n):
2      res = 1
3      for i in range(1, n + 1):
4          res *= i
5      return res
```

Инструкция return может встречаться в произвольном месте функции, ее исполнение завершает работу функции и возвращает указанное значение в место вызова.

Если функция не возвращает значения, то инструкция return используется без возвращаемого значения. В функциях, которым не нужно возвращать значения, инструкция return может отсутствовать.

Локальные и глобальные переменные

Локальные переменные - это переменные созданные в пределах одной области и доступны только там.

Глобальные переменные - это переменные созданы в “глобальной” области вашего кода и доступны как там так и в функциях (без передачи по параметру)

Внутри функции можно использовать переменные, объявленные вне этой функции

```
1  def f():
2      print(a)
3
4  a = 1
5  f()
6
```

Такие переменные (объявленные вне функции, но доступные внутри функции) называются *глобальными*.

Но если инициализировать какую-то переменную внутри функции, использовать эту переменную вне функции не удастся.
Например:

```
1  def f():
2      a = 1
3
4  f()
5  print(a)
6
```

Что тут будет?

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

Если нужно, чтобы функция вернула не одно значение, а два или более, то для этого функция может вернуть список из двух или нескольких значений:

return [a, b]

Тогда результат вызова функции можно будет использовать во множественном присваивании:

n, m = f(a, b)