

Функции — использование встроенных и создание собственных




Олег
Булыгин



Олег Булыгин

IT-аудитор в ПАО Сбербанк

 obulygin91@ya.ru

|  fb.me/obulygin91

План занятия

1. [Что такое функция?](#)
2. [Объявление функций в Python](#)
3. [Параметры функции](#)
4. [Области видимости](#)



Что такое функция?



Что такое функция?

1. В математике

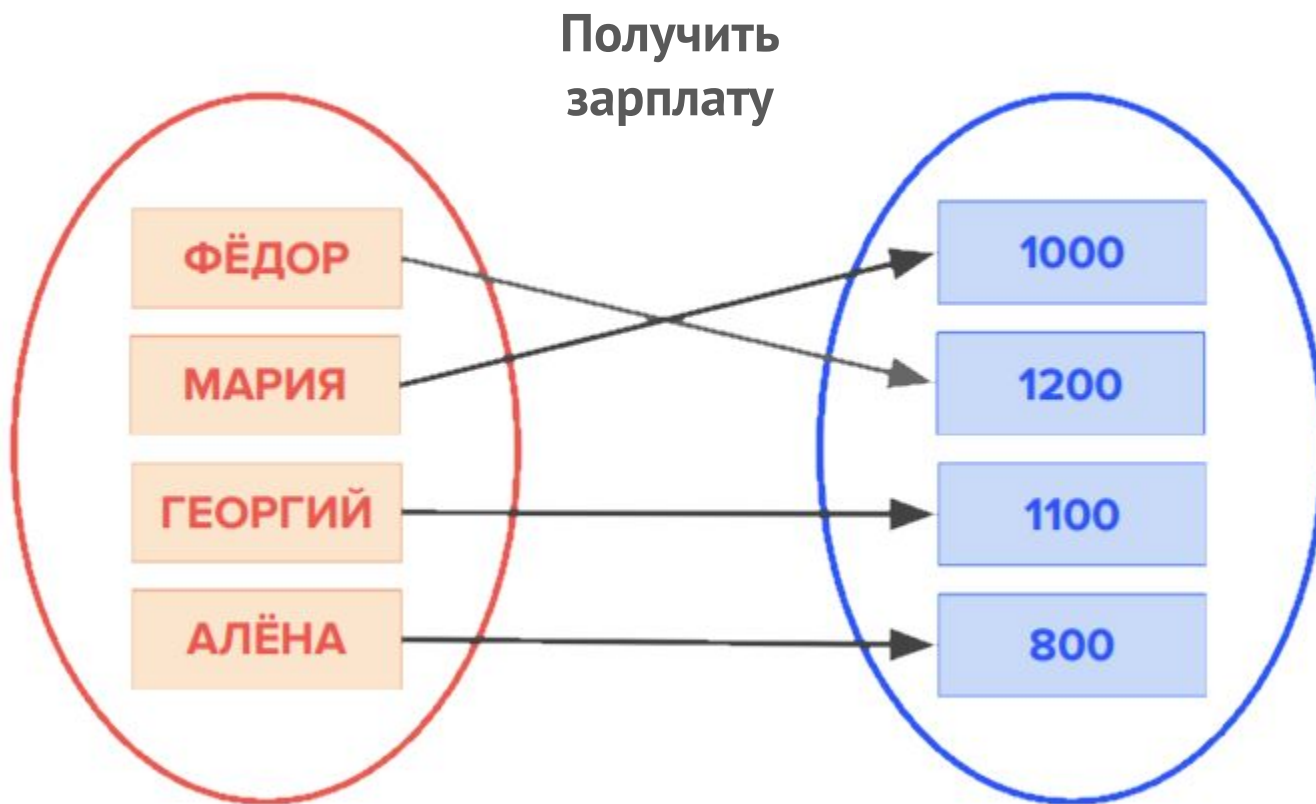
это соответствие между элементами; то как значение одной величины определяет значение другой;

2. В программировании

- а. это обособленный участок кода, который можно вызывать, обратившись к нему по имени, которым он был назван (подпрограмма);
- б. объект, принимающий аргументы и возвращающий значение.

Функции помогают избежать дублирования кода, улучшить его структурированность и читаемость.

Что такое функция?



Что такое функция?

Функция `сходить_в_магазин('магазин', список покупок)`

1. Встать с дивана
2. Найти магазин на карте
3. Доехать до магазина
4. Купить товары по списку


Зафиксировать сумму затрат

`сходить_в_магазин('Десяточка', [молоко, хлеб])`

100 рублей

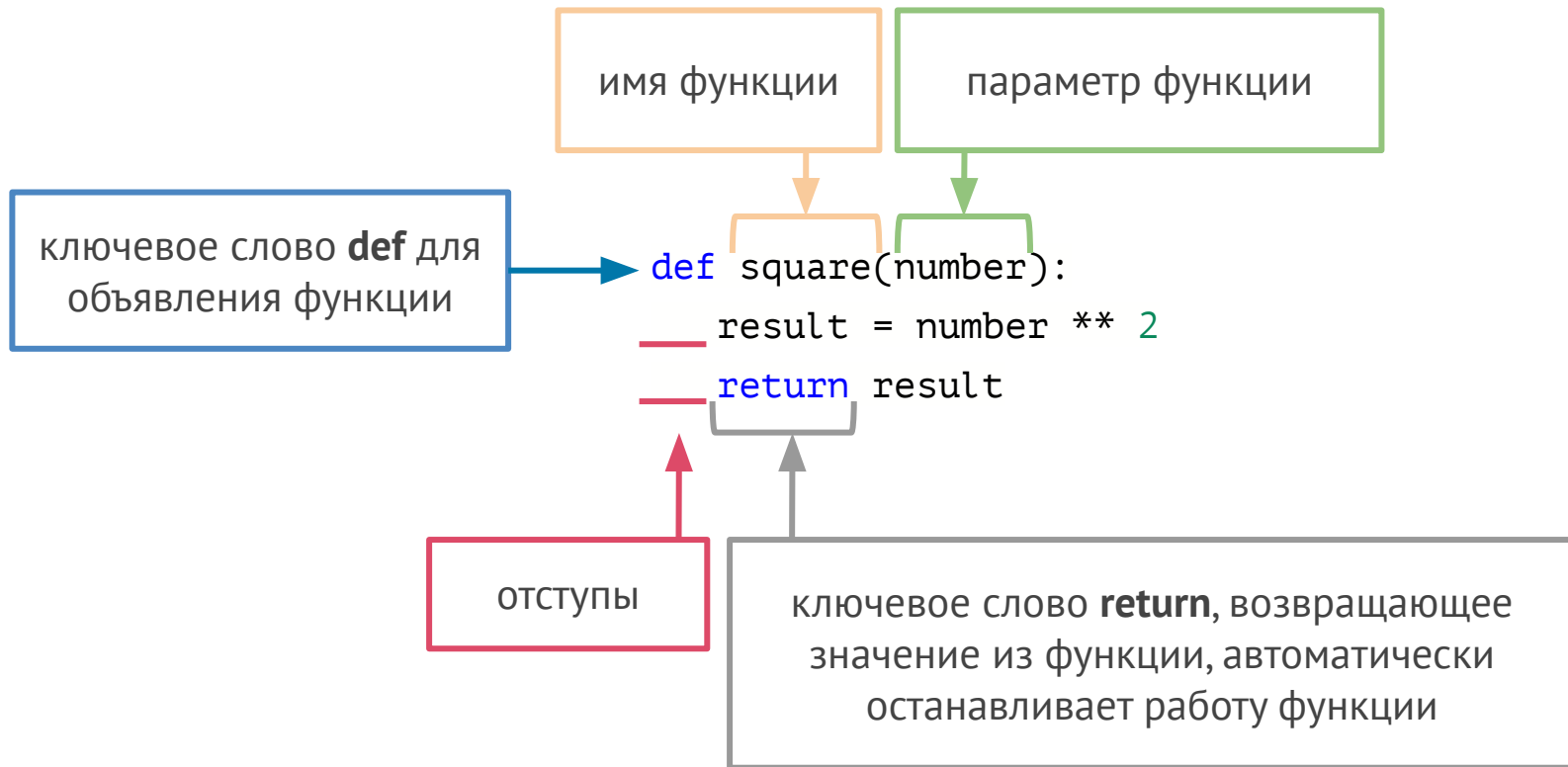
`сходить_в_магазин('DNS', [мышь, клавиатура])`

2000 рублей



Объявление функций в Python

Объявление функций в Python



Функция `help()`

Вызывает справку по нужной функции.



Docstring

(сокр. от *documentation string*, строка документации) встроенное средство документирования модулей, функций, классов и методов.

Сразу после определения указывается строковое значение, которое и будет docstring'ом.

```
def function(a, b):  
    """  
    function(a, b) -> list  
    """  
    return [a, b]
```



Параметры функции



Параметры функции

Функция может принимать более 1 параметра, а может не принимать параметры вообще.

Для всех параметров функций можно указывать значения по-умолчанию, это дает возможность вызвать функцию с меньшим числом параметров.



Тип данных None

`None` – специальный тип данных, который означает отсутствие значения.

Если в функции нет `return`, либо он пустой, то она возвращает `None`.



Области видимости

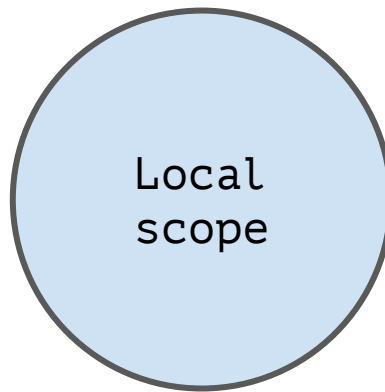
Область видимости

Область видимости (scope) определяет контекст объекта, в рамках которого его можно использовать.

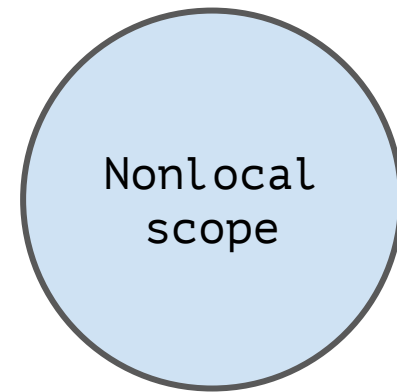
Рассмотрим 3 типа области видимости:



**Глобальная
область видимости**



**Локальная
область видимости**



**Нелокальная
область видимости**




Область видимости

Глобальная область видимости

Глобальный контекст подразумевает, что переменная является глобальной, она определена вне любой из функций и доступна любой функции в программе.

Локальная область видимости

В отличие от глобальных переменных *локальная переменная* определяется внутри функции и доступна только из этой функции, то есть имеет локальную область видимости.



Если Python не может найти нужную переменную в локальной области видимости, то тогда (и только тогда) он будет искать её в области видимости уровня выше.



Операторы `global` и `nonlocal`

Оператор `global` позволяет создать глобальную переменную в локальном контексте.

Оператор `nonlocal` позволяет изменить переменную в области видимости более высокого уровня (которая, в свою очередь, является локальной областью видимости для других переменных).

Анонимные функции

Анонимные функции создаются при помощи инструкции `lambda` и используются для более краткой записи функций с одним выражением.

Выполняются быстрее обычных и не требуют инструкции `return`:

```
lambda x, pow: x**pow
```

Методы

Методы в Python – это функции, которые “принадлежат” к определенному объекту.

У каждого типа объектов есть свои методы.

Примеры методов **списков**:

- `.index()`
- `.count()`
- `.append()`
- `.remove()`
- `.reverse()`

Примеры методов **строк**:

- `.capitalize()`
- `.upper()`
- `.lower()`
- `.replace()`
- `.count()`

Примеры методов **словарей**:

- `.keys()`
- `.values()`
- `.items()`



Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте в чате группы.
- Задачи можно сдавать по частям.
- Зачёт по домашней работе проставляется после того, как приняты **все задачи**.

А чтобы улучшить практические навыки программирования, выполните **задания в тренажёре** в личном кабинете

**Задавайте вопросы и
пишите отзыв о лекции!**

Олег Булыгин



obulygin91@ya.ru



fb.me/obulygin91