Функции Python

объект, принимающий аргументы и возвращающий значение

Определить функцию

```
def add(x, y):
    return x + y
```

Функция может принимать произвольное количество аргументов или не принимать их вовсе

Функция также может принимать переменное количество позиционных аргументов, тогда перед именем ставится *:

```
def func(*args):
  return args
```

Функция может принимать и произвольное число именованных аргументов, тогда перед именем ставится **:

```
def func(**kwargs):
  return kwargs
```

Аннотация функции
позволяет явно прописать тип данных *параметров* и *возвращаемых значений*

```
def f(x: float = 3.5) -> int:
  return int(x)
```

Анонимные функции lambda

lambda функции, в отличие от обычной, не требуется инструкция return

`func = lambda x, y: x + y`

Встроенные функции

Все встроенные объекты можно посмотреть следующим образом:

```
for ein __builtins__.__dict__:
    print(e)
```

Посмотреть подробную документацию по функции можно командой `print(help("range"))`

▼ Общие функции

- `print` — функция вывода (например в консоль)
- `range` — вернет последовательность чисел, с 0 до N (с шагом на 1 по умолчанию)
- `dir` — список имен объекта (если он указан) или список объектов доступных в локальной области (если объект не указан)
- `divmod` — вернет частное и остаток от деления двух чисел
- `enumerate` — вернет объект, который генерирует кортежи из двух элементов (индекса и самого элемента)
- `format` — форматирование (например форматирование строки)
- `globals` — глобальные имена (в виде словаря)
- `locals` — локальные имена (в виде словаря)
- `help` — вызовет встроенную справку
- `input` — вернет введенную пользователем строку
- `open` — откроет файл и вернет его объект

▼ Преобразование типов

- `type` — вернет тип объекта
- `str` — преобразование в строку

- `int` — преобразование в число
- `float` — преобразование в число с плавающей точкой
- `complex` — преобразование в комплексное число
- `bool` — преобразование к булевому типу
- `tuple` — преобразование к кортежу
- `dict` — преобразование к словарю
- `frozenset` — приведение к неизменяемому множеству
- `list` — приведение к списку
- `set` — преобразование к множеству
- `slice` — создание среза
- `bin` — приведение целого числа к двоичной строке
- `hex` — целое число в шестнадцатеричную строку
- `oct` — целое число в восьмеричную строку

▼ Математика

- `pow` — возводит число в степень
- `abs` — возвращает модуль числа
- `round` — округление до указанного количества знаков после запятой

▼ Итерируемые объекты

- `all` — если все элементы итерируемого объекта истинные, вернет True
- `any` — если хотя бы один элемент итерируемого объекта истинный, вернет True
- `iter` — возвращает объект итератора
- `next` — возвращает следующий элемент итератора
- `zip` — позволяет пройтись одновременно по нескольким итерируемым объектам
- `len` — выводит количество элементов в объекте (списке, строке и т.д.)
- `filter` — фильтрация элементов переданной последовательности
- `map` — применяет указанную функцию к каждому элементу указанной последовательности
- `min` — вернет минимальный элемент последовательности
- `max` — вернет максимальный элемент последовательности
- `reversed` — вернет обратный итератор по указанной последовательности
- `sorted` — вернет новый отсортированный список

- `sum` — вернет сумму элементов последовательности

▼ Работа со строковыми символами

- `ascii` — возвращает строковое представление объекта и заменяет не-ASCII символы на экранированные последовательности
- `chr` — возвращает символ по числовому представлению
- `ord` — возвращает код символа

▼ Работа с байтами

- `bytes` — преобразование в тип bytes
- `bytearray` — преобразование к bytearray
- `memoryview` — создает объект memoryview

▼ Исполнение кода

- `exec` — динамически исполняет программный код
- `eval` — выполняет строку программного кода
- `compile` — компилирует исходный код в объект кода, который после можно выполнить с помощью `eval` или `exec`

▼ Объекты и классы

- `object` — возвращает базовый объект
- `id` — возвращает идентификатор указанного объекта
- `hash` — возвращает хэш объекта
- `isinstance` — если объект является экземпляром указанного класса или его подклассом, вернет True
- `issubclass` — если класс является подклассом другого класса, вернет True
- `callable` — если объект поддерживает вызов, вернет True
- `classmethod` — представляет указанную функцию методом класса
- `repr` — возвращает строковое представление указанного объекта
- `setattr` — устанавливает аттрибут объекта
- `getattr` — извлечение значения атрибута объекта
- `hasattr` — проверяет, имеет ли объект указанный аттрибут
- `delattr` — удаление аттрибута
- `staticmethod` — представляет указанную функцию статичным методом
- `super` — дает возможность использования методов класса-родителя в классе потомке

- `vars` — вернет словарь из аттрибутов объекта