

# Функция `isinstance()` в Python, принадлежность экземпляра к классу

[Справочник по языку Python3.](#) / [Встроенные функции Python](#) / Функция `isinstance()` в Python, принадлежность экземпляра к классу

## Позволяет проверить принадлежность экземпляра к классу

### Синтаксис:

```
isinstance(object, classinfo)
```

### Параметры:

- `object` - объект, требующий проверки,
- `classinfo` - [класс](#), [кортеж](#) с классами или рекурсивный кортеж кортежей или с версии Python 3.10 может быть объединением нескольких типов (например `int | str`).

### Возвращаемое значение:

- `bool`.

### Описание:

Функция [isinstance\(\)](#) вернет `True`, если проверяемый объект `object` является **экземпляром** любого указанного в `classinfo` класса (классов) или его подкласса (прямого, косвенного или виртуального).

Если объект `object` не является экземпляром данного типа, то функция всегда возвращает `False`.

Функцией `isinstance()` можно проверить класс, [кортеж](#) с классами (с Python 3.10 может быть записью, объединяющей нескольких типов, например, `int | str`), либо рекурсивный кортеж кортежей. Другие типы последовательностей аргументом `classinfo` не поддерживаются.

Если аргумент `classinfo` не является классом, либо кортежем с классами (с Python 3.10 может быть записью, объединяющей нескольких типов, например, `int | str`), то возбуждается [исключение `TypeError`](#).

*Изменено в Python 3.10: аргумент `classinfo` может иметь [`mun Union`](#) и записываться как побитовое или - `|`.*

```
>>> isinstance(1, int | str)
# True
>>> isinstance("", int | str)
# True
```

Доступ к пользовательскому типу для объекта `union` можно получить из `types.UnionType` и использовать для проверок [isinstance\(\)](#). Невозможно создать экземпляр объекта из типа:

```
>>> import types
>>> isinstance(int | str, types.UnionType)
# True
>>> types.UnionType()
# Traceback (most recent call last):
#   File "<stdin>", line 1, in <module>
# TypeError: cannot create 'types.UnionType' instances
```

## Примеры проверок принадлежности экземпляра к классу.

Взгляните на следующий код:

```
>>> a = 3
>>> isinstance(a, object)
# True
>>> type(a) == object
# False
```

```
>>> isinstance(type(a), object)
# True
```

Молодые программисты часто не понимают что здесь происходит. Давайте разбираться...

Функция `type()` с одним аргументом `object` возвращает тип объекта - **точный класс**, из которого был создан переданный аргумент `object`. Мы знаем, что все [целые числа в Python](#) являются типом `int`. Это означает, что проверка объекта `type(3) == object` в точности эквивалентна проверке объекта `int == object`, что однозначно неверно.

В Python, как это известно - все является объектом, следовательно дочерний класс (в данном случае [int](#)) наследуется от родительского (в данном случае `object`) и считается, что такое поведение имеет отношение `is a`. Отношение `is a` - это то, что проверяет [функция `isinstance\(\)`](#).

Существует аналогичная [функция `issubclass\(\)`](#) для проверки того же отношения, только **для класса** вместо **экземпляра этого класса**. В большинстве случаев `isinstance(x, y) == issubclass(type(x), y)`.

Классы всегда имеют одну и ту же ссылку в одной сессии интерпретатора, поэтому можно использовать [оператор `is`](#) вместо оператора `==` для сравнения. Таким образом, `type(3) is int` будет истинным.

Изобразим сказанное выше на примере:

```
class Base:
    def __init__(self):
        self.foo='foo'

class Sub(Base):
    def __init__(self):
        super().__init__()

>>> obj = Base()
>>> sub_obj = Sub()
>>> isinstance(obj, Base)
# True
>>> isinstance(obj, Sub)
# False
>>> isinstance(sub_obj, Base)
# True
>>> isinstance(sub_obj, Sub)
# True
>>> type(sub_obj) == Base
# False
>>> type(sub_obj)
# <class '__main__.Sub'>
>>> type(sub_obj) == Sub
# True
>>> issubclass(type(sub_obj), Base)
# True
```

## Общие приемы проверок.

```
>>> x = 1
>>> isinstance(x, int)
# True
>>> x = [1, 2, 3]
>>> isinstance(x, list)
# True
>>> x = (1, 2, 3)
>>> isinstance(x, tuple)
# True

# Проверим, является ли строка 'Hello' одним из типов, описанных в параметре type
>>> isinstance('Hello', (float, int, str, list, dict, tuple))
# True

# Проверка, на принадлежность к экземпляром myObj
class myObj:
```

```
name = "John"

y = myObj()
>>> isinstance(y, myObj)
# True
```

Содержание раздела:

- [ОБЗОРНАЯ СТРАНИЦА РАЗДЕЛА](#)
- [Функция abs\(\), абсолютное значение числа](#)
- [Функция all\(\), все элементы True](#)
- [Функция any\(\), хотя бы один элемент True](#)
- [Функция ascii\(\), преобразует строку в ASCII](#)
- [Функция bin\(\), число в двоичную строку](#)
- [Класс bool\(\), логическое значение объекта](#)
- [Функция breakpoint\(\), отладчик кода](#)
- [Класс bytearray\(\), преобразует в массив байтов](#)
- [Класс bytes\(\), преобразует в строку байтов](#)
- [Функция callable\(\), проверяет можно ли вызвать объект](#)
- [Функция chr\(\), число в символ Юникода](#)
- [Класс classmethod, делает функцию методом класса](#)
- [Функция compile\(\) компилирует блок кода Python](#)
- [Класс complex\(\), преобразует в комплексное число](#)
- [Функция delattr\(\), удаляет атрибут объекта](#)
- [Класс dict\(\) создает словарь](#)
- [Функция dir\(\), все атрибуты объекта](#)
- [Функция divmod\(\), делит числа с остатком](#)
- [Функция enumerate\(\), счетчик элементов последовательности](#)
- [Функция eval\(\), выполняет строку-выражение с кодом](#)
- [Функция exec\(\), выполняет блок кода](#)
- [Функция filter\(\), фильтрует список по условию](#)
- [Класс float\(\), преобразует в вещественное число](#)
- [Функция format\(\), форматирует значение переменной](#)
- [Класс frozenset\(\), преобразует в неизменяемое множество](#)
- [Функция getattr\(\), значение атрибута по имени](#)
- [Функция globals\(\), переменные глобальной области](#)
- [Функция hasattr\(\), наличие атрибута объекта](#)
- [Функция hash\(\), хэш-значение объекта](#)
- [Функция help\(\), справка по любому объекту](#)
- [Функция hex\(\), число в шестнадцатеричную строку](#)
- [Функция id\(\), идентификатор объекта](#)
- [Функция input\(\), ввод данных с клавиатуры](#)
- [Класс int\(\), преобразует в тип int](#)
- [Функция isinstance\(\), принадлежность экземпляра к классу](#)
- [Функция issubclass\(\), проверяет наследование класса](#)
- [Функция iter\(\), создает итератор](#)
- [Функция len\(\), количество элементов объекта](#)
- [Класс list\(\)](#)
- [Функция locals\(\), переменные локальной области](#)
- [Функция map\(\), обработка последовательности без цикла](#)
- [Функция max\(\), максимальное значение элемента](#)
- [Класс memoryview\(\), ссылка на буфер обмена](#)
- [Функция min\(\), минимальное значение элемента](#)
- [Функция next\(\), следующий элемент итератора](#)

- [Класс object\(\)](#), возвращает безликий объект
- [Функция oct\(\)](#), число в восьмеричную строку
- [Функция open\(\)](#), открывает файл на чтение/запись
- [Функция ord\(\)](#), число символа Unicode
- [Функция pow\(\)](#), возводит число в степень
- [Функция print\(\)](#), печатает объект
- [Класс property\(\)](#), метод класса как свойство
- [Класс range\(\)](#), генерирует арифметические последовательности
- [Функция repr\(\)](#), описание объекта
- [Функция reversed\(\)](#), разворачивает последовательность
- [Функция round\(\)](#), округляет число
- [Класс set\(\)](#), создает или преобразовывает в множество
- [Функция setattr\(\)](#), создает атрибут объекта
- [Класс slice\(\)](#), шаблон среза
- [Функция sorted\(\)](#), выполняет сортировку
- [Декоратор staticmethod\(\)](#), метод класса в статический метод
- [Класс str\(\)](#), преобразует объект в строку
- [Функция sum\(\)](#), сумма последовательности
- [Функция super\(\)](#), доступ к унаследованным методам
- [Класс tuple\(\)](#), создает или преобразует в кортеж
- [Класс type\(\)](#), возвращает тип объекта
- [Функция vars\(\)](#), словарь переменных объекта
- [Функция zip\(\)](#), объединить элементы в список кортежей
- [Функция \\_\\_import\\_\\_\(\)](#), находит и импортирует модуль
- [Функция aiter\(\)](#), создает асинхронный итератор
- [Функция anext\(\)](#), следующий элемент асинхронного итератора

ХОЧУ ПОМОЧЬ  
ПРОЕКТУ

[DOCS-Python.ru](#)™, 2024 г.

(Внимание! При копировании материала ссылка на источник обязательна)

[@docs\\_python\\_ru](#)