

Основы языка программирования Python¹

Модули

```
# application.py
```

```
import tools
```

```
foo = tools.add(10, 20)
```

```
bar = tools.mul(4, 2)
```

```
print(f"{tools=}")
```

```
print(f"{type(tools)=}")
```

```
print(f"{dir(tools)=}")
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

tools=<module 'tools' from '../tools.py'>

type(tools)=<class 'module'>

dir(tools)=['__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__spec__', 'add', 'mul']

```
# application.py
```

```
import tools as tl
```

```
foo = tl.add(10, 20)
```

```
bar = tl.mul(4, 2)
```

```
print(f"{tl=}")
```

```
print(f"{type(tl)=}")
```

```
print(f"{dir(tl)=}")
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
tl=<module 'tools' from '.../tools.py'>
```

```
type(tl)=<class 'module'>
```

```
dir(tl)=['__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__',  
        '__package__', '__spec__', 'add', 'mul']
```

```
# application.py
```

```
from tools import add, mul
```

```
foo = add(10, 20)
```

```
bar = mul(4, 2)
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
# application.py
```

```
from math import sqrt
from cmath import sqrt as csqrt
```

```
from tools import add, mul
```

```
foo = sqrt(add(10, 20))
bar = csqrt(mul(4, -2))
```

```
# tools.py
```

```
def add(a, b):
    return a + b
```

```
def mul(a, b):
    return a * b
```

```
# application.py
```

```
from tools import *
```

```
foo = add(10, 20)
```

```
bar = mul(4, 2)
```



```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
# application.py
```

```
print("Внутри application.py - 1")
```

```
import tools
```

```
print("Внутри application.py - 2")
```

```
foo = tools.add(10, 20)
```

```
bar = tools.mul(4, 2)
```

```
print("Внутри application.py - 3")
```

```
# tools.py
```

```
print("Внутри tools.py - 1")
```

```
def add(a, b):  
    return a + b
```

```
print("Внутри tools.py - 2")
```

```
def mul(a, b):  
    return a * b
```

```
> python application.py
```

```
Внутри application.py - 1
```

```
Внутри tools.py - 1
```

```
Внутри tools.py - 2
```

```
Внутри application.py - 2
```

```
Внутри application.py - 3
```

```
# application.py
```

```
import tools
```

```
foo = tools.add(10, 20)
```

```
bar = tools.mul(4, 2)
```

```
print("Внутри application.py:", f"__name__={}")
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
print("Внутри tools.py:", f"__name__={}")
```

```
> python application.py
```

```
Внутри tools.py: __name__='tools'
```

```
Внутри application.py: __name__='__main__'
```



```
# application.py
```

```
import tools
```

```
if __name__ == "__main__":  
    foo = tools.add(10, 20)  
    bar = tools.mul(4, 2)  
    print("Внутри application.py:", f"__name__={__name__}")
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
if __name__ == "__main__":  
    print("Внутри tools.py:", f"__name__={__name__}")
```

```
> python application.py
```

```
Внутри application.py: __name__='__main__'
```

```
# application.py
```

```
import tools
```

```
if __name__ == "__main__":  
    foo = tools.add(10, 20)  
    bar = tools.mul(4, 2)  
    print("Внутри application.py:", f"__name__={__name__}")  
    print("Внутри application.py:", f"__file__={__file__}")  
    print("Внутри application.py:", f"tools.__file__={tools.__file__}")
```

```
# tools.py
```

```
def add(a, b):  
    return a + b
```

```
def mul(a, b):  
    return a * b
```

```
print("Внутри tools.py:", f"__name__={__name__}")  
print("Внутри tools.py:", f"__file__={__file__}")
```

```
> python application.py
```


```
Внутри tools.py:      __name__='tools'  
Внутри tools.py:      __file__='.../tools.py'  
Внутри application.py: __name__='__main__'  
Внутри application.py: __file__='.../application.py'  
Внутри application.py: tools.__file__='.../tools.py'
```

Пакеты модулей

```
src/11. Modules/example_modules_10.py  
src/11. Modules/example_modules_11.py  
src/11. Modules/example_modules_12.py
```

Установка библиотек с помощью pip

https://pypi.org



[Помощь](#) [Спонсоры](#) [Войти](#) [Зарегистрироваться](#)

Фильтр по классификатору 10 000+ проектов по запросу «numpy» Сортировать по

Соответствию

Framework

Topic

Development Status

License

Programming Language


Operating System


Environment


Intended Audience


Natural Language


Typing


 **numpy** 16 мар. 2025 г.
Fundamental package for array computing in Python

 **numpy-2425** 1 апр. 2025 г.
A custom NLP package

 **video2numpy** 11 сент. 2023 г.
Video reading into numpy

 **wandb2numpy** 7 дек. 2022 г.
Library to export data from WandB to numpy arrays or csv files.

 **x256numpy** 16 янв. 2025 г.
Quickly find the nearest xterm 256 color index of an RGB (NumPy Version)

 **lapjv-numpy2** 19 июн. 2024 г.
Linear sum assignment problem solver using Jonker-Volgenant algorithm. Fork to support numpy 2.x

Установка библиотек с помощью pip

14

Установка пакетов для всех пользователей ОС
(требуется права администратора):

```
pip install пакет_1 пакет_2 ... пакет_N
```

ИЛИ

```
python -m pip install пакет_1 пакет_2 ... пакет_N
```

Установка библиотек с помощью pip

15

Установка пакетов для текущего пользователя ОС
(права администратора не требуются):

```
pip install --user пакет_1 пакет_2 ... пакет_N
```

ИЛИ

```
python -m pip install --user пакет_1 пакет_2 ... пакет_N
```

Получение списка установленных библиотек с помощью pip

16

Получить список всех установленных пакетов:

```
pip list
```

Получить список установленных пакетов, для которых
появились новые версии:

```
pip list --outdated
```


Обновление библиотек с помощью pip

Обновление библиотек для всех пользователей ОС
(требуется права администратора):

```
pip install --upgrade пакет_1 пакет_2 ... пакет_N
```

Обновление библиотек для текущего пользователя ОС
(права администратора не требуются):

```
pip install --upgrade --user пакет_1 пакет_2 ... пакет_N
```

Удаление библиотек с помощью `pip`

```
pip uninstall пакет_1 пакет_2 ... пакет_N
```

Установка библиотек для инженерных и научных вычислений

19

```
pip install --user numpy matplotlib scipy pandas
```