

Погружение в Python

Урок 6
Модули



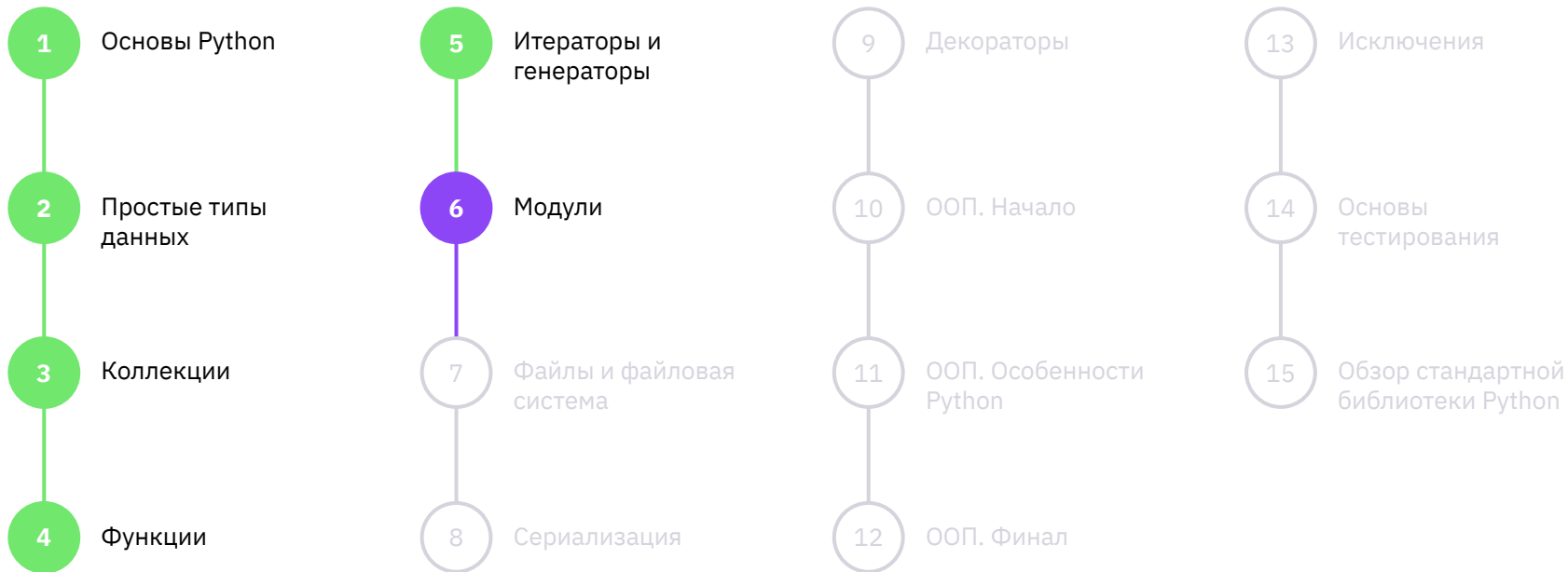


Содержание урока





План курса



Что будет на уроке сегодня

- 📌 Разберём работу с модулями в Python
- 📌 Изучим особенности импорта объектов в проект
- 📌 Узнаем о встроенных модулях и возможностях по созданию своих модулей и пакетов
- 📌 Разберём модуль `random` отвечающий за генерацию случайных чисел





Ещё раз про import





import module_name

Строки импорта рекомендуется писать в самом начале файла, оставляя 1-2 пустые строки после него.

```
import sys
```

```
print(sys)
```

```
print(sys.builtin_module_names)
```

```
print(*sys.path, sep='\n')
```



Переменная `sys.path`

Содержимое переменной `sys.path` формируется динамически.



PYTHONPATH - переменная с путями до мест расположения модулей.

Давайте рассмотрим антипримеры импорта.





Использование from и as

Расширенные возможности импорта.

1

import from

```
from sys import  
builtin_module_names
```

2

import as

```
import numpy as np
```



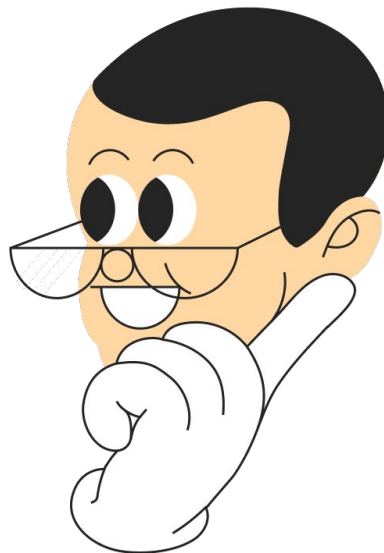


Плохой `import *` (импорт звёздочка)

Разберём на примерах.

✓ `from имя_модуля import *`

Подобная запись импортирует из модуля все глобальные объекты за исключением тех, чьи имена начинаются с символа подчёркивания.

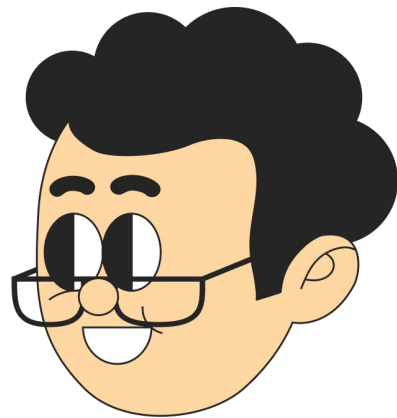




Переменная `__all__`

✓ `__all__ = ['func', '_secret', '...']`

Список имён объектов, заключённых в кавычки, т.е. строки для импорта через «звёздочку».





Перед вами несколько строк кода.
Какие переменные будут доступны после
импорта для работы в основном файле?

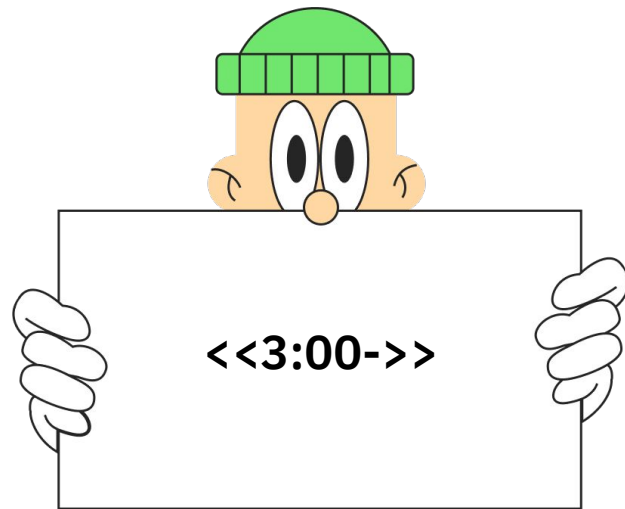
У вас 3 минуты.





import

```
import sys
from random import *
from super_module import func as f
```





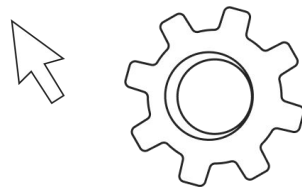
Виды модулей





Встроенные модули

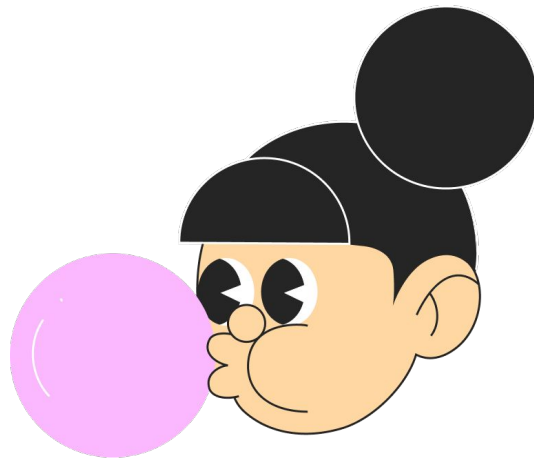
- ✓ Стандартная библиотека устанавливается вместе с интерпретатором. Дополнительные манипуляции по установке не требуются. Всё работает «из коробки».
- ✓ Для использования модуля стандартной библиотеки достаточно его импортировать в ваш код.
- ✓ Большинство частых задач легко решаются средствами стандартной библиотеки. Достаточно обратиться к справке и найти нужный модуль.
- ✓ Некоторые модули стандартной библиотеки разрабатывались настолько давно, что не отвечают современным требованиям решения задач. В таком случае на помощь приходят внешние решения. И наоборот. Каждое обновление Python вносит улучшения в библиотеку, зачастую более эффективные, чем внешние решения.





Свои модули

- ✓ Документация по модулю в виде многострочного комментария (три пары двойных кавычек),
- ✓ Импорт необходимых пакетов, модулей, классов, функций и т.п. объектов,
- ✓ Определение констант уровня модуля,
- ✓ Создание классов модуля при ООП подходе,
- ✓ Создание функций модуля,
- ✓ Определение переменных модуля,
- ✓ Покрытие тестами, если оно не вынесено в отдельный пакет,
- ✓ Main код.





Пишем свой модуль: `__name__ == '__main__'`

Магическая переменная `__name__` содержит полное имя импортируемого модуля или `'__main__'`

```
...  
if __name__ == '__main__':  
    print('Starting main code')  
...
```





Разбор плохого импорта



x = base_math.mul

Передача имени в другую переменную



y = base_math._START_MULT

Обращение к защищённой или приватной переменной



Создание пакетов и их импорт



Файл `__init__.py`

Директоия с `__init__.py` превращается в пакет





Разница между модулем и пакетом

Пакет хранит модули.

- ✓ Пакет — директория с `__init__.py` файлом
- ✓ Пакет можно импортировать как модуль
- ✓ Внутри файл `__init__.py` можно прописать код, который будет выполняться при импорте пакета.

```
sound/
__init__.py
formats/
    __init__.py
    wavread.py
    wavwrite.py
    aiffread.py
    aiffwrite.py
    auread.py
    auwrite.py
    ...
effects/
    __init__.py
    echo.py
    surround.py
    reverse.py
    ...
filters/
    __init__.py
    equalizer.py
    vocoder.py
    karaoke.py
    ...
```

Top-level package
Initialize the sound package
Subpackage for file format conversions

Subpackage for sound effects

Subpackage for filters



Варианты импорта

Относительный импорт запрещён в основном файле проекта.



Простой импорт

```
import mathematical
```



Абсолютный импорт

```
from mathematical import  
base_math as bm
```

```
from  
mathematical.advanced_math  
import exp
```



Относительный импорт

```
from . import other_module
```

```
from .. import other_module
```

```
from ..other_package import  
other_module
```



Перед вами список файлов проекта.
Напишите в чат какие файлы надо удалить
или переименовать, а какие верные.

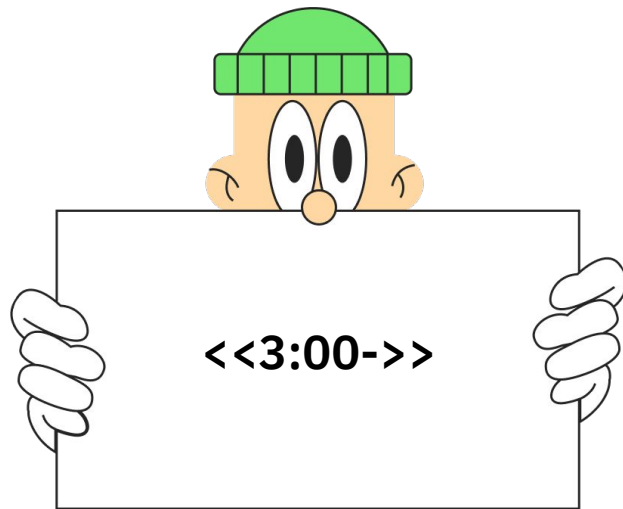
У вас 3 минуты.





Модули

```
__all__.py  
__init__.py  
__main__.py  
init.py  
math.py  
random.py
```





Некоторые модули
«из коробки»





Модуль sys и запуск скрипта с параметрами

Модуль sys обеспечивает доступ к некоторым переменным, используемым или поддерживаемым интерпретатором, а также к функциям, тесно взаимодействующим с интерпретатором.

1

Код файла:

```
from sys import argv
```

```
print('start')
```

```
print(argv)
```

```
print('stop')
```

2

Команда запуска:

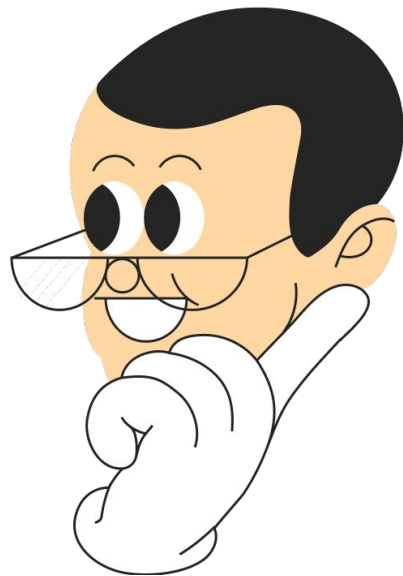
```
python script.py -d 42 -s "Hello world!" -k 100
```




Модуль random

Модуль используется для генерации псевдослучайных чисел.

- ✓ `random()` — генерирует псевдослучайные числа в диапазоне $[0, 1)$
- ✓ `seed(a=None, version=2)` — инициализирует генератор. Если значение `a` не указано, для инициализации используется текущее время ПК
- ✓ `getstate()` — возвращает объект с текущим состоянием генератора
- ✓ `setstate(state)` — устанавливает новое состояние генератора, принимая на вход объект, возвращаемый функцией `getstate`





Несколько часто используемых функции random



randint(a, b)

целое число от a до b



randrange(start, stop[, step])

число из диапазона



uniform(a, b)

вещественное число от a до b



shuffle(x)

перемешиваем коллекцию x in place



choice(seq)

случайный элемент последовательности



sample(population, k, *, counts=None)

Выборка в k элементов из population



Перед вами несколько строк
кода. Напишите в чат что они вернут
не запуская программу.

У вас 3 минуты.

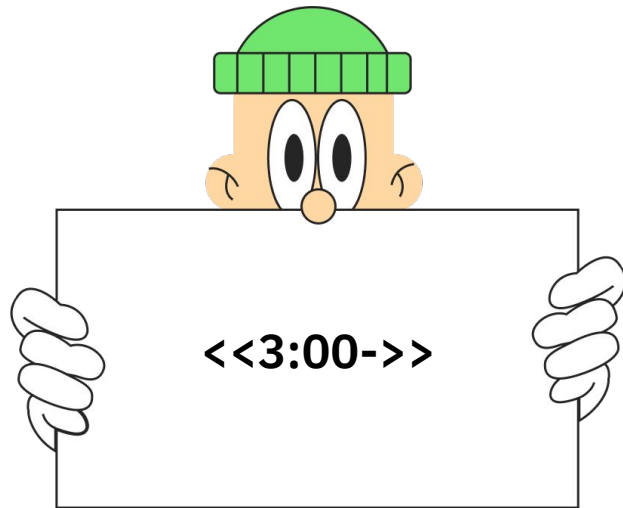


Некоторые модули «из коробки»

```
import random
from sys import argv

print(random.uniform(int(argv[1]), int(argv[2])))
print(random.randrange(int(argv[1]), int(argv[2]),
int(argv[1])))
print(random.sample(range(int(argv[1]),
int(argv[2]), int(argv[1])), 10))
```

Скрипт запущен командой: `python3 main.py 10 1010`









Итоги занятия



На этой лекции мы

-  Разобрали работу с модулями в Python
-  Изучили особенности импорта объектов в проект
-  Узнали о встроенных модулях и возможностях по созданию своих модулей и пакетов
-  Разобрали модуль random отвечающий за генерацию случайных чисел

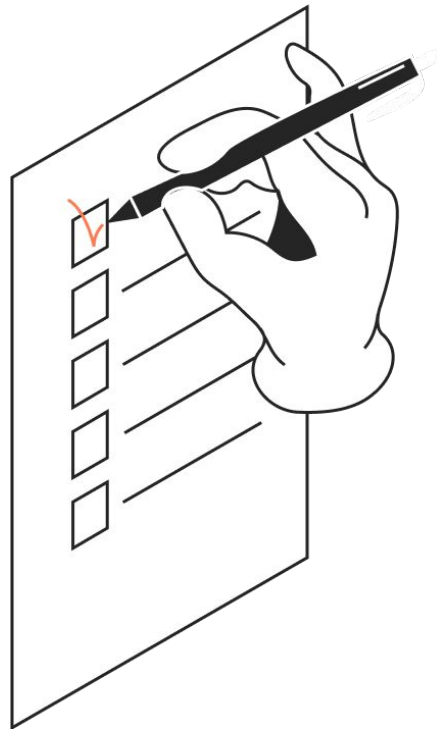




Задание

Соберите файлы прошлых уроков в отдельные директории и сделайте их пакетами. Измените сами файлы, чтобы они были импортируемыми модулями.

Подсказка: используйте файлы `__init__.py` и проверку переменной `__name__`.





Спасибо за внимание