Python Modules

https://docs.python.org/3/tutorial/modules.html

Считайте модуль таким же, как и библиотеку кода. Файл, содержащий набор функций, которые вы хотите включить в свое приложение.
Чтобы создать модуль, просто сохраните нужный код в файле с расширением `.py`:

```
def greeting(name):
  print("Hello, " + name)
```

Теперь мы можем использовать только что созданный модуль с помощью `import` оператора:

```
import mymodule
mymodule.greeting("Jonathan")
```

- Модуль может содержать функции, как уже описано, а также переменные всех типов (массивы, словари, объекты и т. д.):
  - Сохраните этот код в файле mymodule.py

```
person1 = {
  "name": "John",
  "age": 36,
  "country": "Norway"
}
```

  - Импортируйте модуль с именем mymodule и получите доступ к словарю person1:

```
import mymodule
a = mymodule.person1["age"]
print(a)
```

- Вы можете создать псевдоним при импорте модуля, используя `as` ключевое слово:

```
import mymodule as mx
a = mx.person1["age"]
print(a)
```

Built-in Modules

В Python есть несколько встроенных модулей, которые вы можете импортировать в любое время.
Импортируйте и используйте `platform` модуль:

```
import platform
x = platform.system()
print(x)
```

Существует встроенная функция для вывода списка всех имен функций (или имен переменных) в модуле. Функция `dir()`:

```
import platform
x = dir(platform)
print(x)
```

Функцию dir() можно использовать для *всех* модулей, в том числе и для тех, которые вы создаете сами.

Import From Module

Вы можете выбрать импорт только частей из модуля, используя `from` ключевое слово.

EXAMPLE:

- Названный модуль `mymodule` имеет одну функцию и один словарь:

```
def greeting(name):
  print("Hello, " + name)
person1 = {
  "name": "John",
  "age": 36,
  "country": "Norway"
}
```

Импортируйте из модуля только словарь person1:

```
from mymodule import person1
print (person1["age"])
```

При импорте с использованием `from` ключевого слова не используйте имя модуля при ссылке на элементы в модуле. Пример: `person1["age"]`, **не** ~~`mymodule.person1["age"]`~~

argparse — Анализатор параметров командной строки, аргументов и подкоманд

https://docs.python.org/3/library/argparse.html

Модуль `argparse` упрощает написание удобных интерфейсов командной строки. Программа определяет, какие аргументы ей требуются, и `argparse` выясняет, как их анализировать из файлов `sys.argv`. Модуль `argparse` также автоматически генерирует справку и сообщения об использовании и выдает ошибки, когда пользователи передают программе недопустимые аргументы.

math — Математические функции

Этот модуль обеспечивает доступ к математическим функциям.

https://docs.python.org/3/library/math.html

json — Кодировщик и декодер JSON

https://docs.python.org/3/library/json.html

os— Разные интерфейсы операционной системы

Этот модуль предоставляет портативный способ использования функций, зависящих от операционной системы.

https://docs.python.org/3/library/os.html

sys— Системные параметры и функции

Этот модуль обеспечивает доступ к некоторым переменным, используемым или поддерживаемым интерпретатором, а также к функциям, тесно взаимодействующим с интерпретатором. Он всегда доступен.
https://docs.python.org/3/library/sys.html

pdb— Отладчик Python

Модуль `pdb` определяет интерактивный отладчик исходного кода для программ Python. Он поддерживает установку (условных) точек останова и пошаговое выполнение на уровне строки исходного кода, проверку кадров стека, листинг исходного кода и оценку произвольного кода Python в контексте любого кадра стека.
https://docs.python.org/3/library/pdb.html

Python click

Click — это пакет Python для создания красивых интерфейсов командной строки компонуемым способом с минимальным количеством кода. Это «Комплект для создания интерфейса командной строки». Он легко настраивается, но поставляется с разумными настройками по умолчанию.
Он направлен на то, чтобы сделать процесс написания инструментов командной строки быстрым и увлекательным, а также предотвратить любое разочарование, вызванное невозможностью реализовать предполагаемый API CLI.

asyncio— Асинхронный ввод-вывод

asyncio — это библиотека для написания параллельного кода с использованием синтаксиса async/await. asyncio используется в качестве основы для нескольких асинхронных фреймворков Python, которые обеспечивают высокопроизводительные сетевые и веб-серверы, библиотеки подключения к базам данных, распределенные очереди задач и т. д.
asyncio часто идеально подходит для кода, связанного с вводом-выводом, и высокоуровневого **структурированного** сетевого кода.