GeekBrains

Погружение в Python

Урок 8
Сериализация

Содержание урока

План курса

1. Основы Python
2. Простые типы данных
3. Коллекции
4. Функции
5. Итераторы и генераторы
6. Модули
7. Файлы и файловая система
8. Сериализация

Что будет на уроке сегодня

- 📌 Разберёмся в сериализации и десериализации данных
- 📌 Изучим самый популярный формат сериализации — JSON
- 📌 Узнаем о чтении и записи таблиц в формате CSV
- 📌 Разберёмся с внутренним сериализатором Python — модулем pickle

Сериализация данных

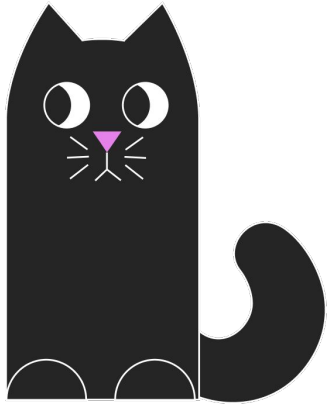
Сериализация данных

1

Сериализация — это процесс преобразования объекта в поток байтов для сохранения или передачи в память, базу данных или файл.

2

Десериализация – восстановление объектов из байт, сохранение которых было произведено ранее. Процедура выгрузки «зафиксированной» информации пользователем.



JSON (JavaScript Object Notation)

Формат JSON

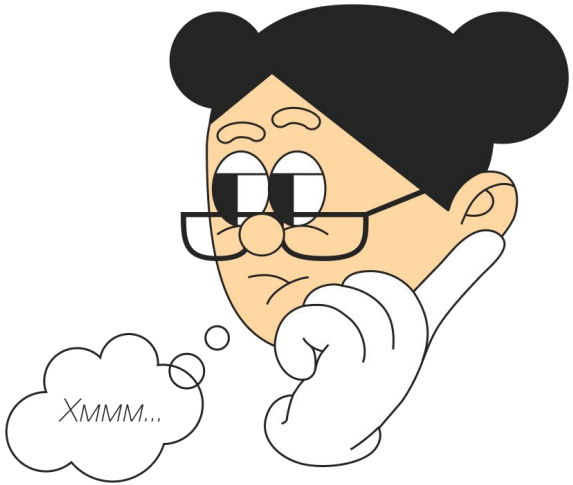
JSON похож на словарь Python

Python	JSON	Python
dict	object	dict
list, tuple	array	list
str	string	str
int	number (int)	int
float	number (float)	float
True	true	True
False	false	False
None	null	None

Преобразование JSON в Python

import json

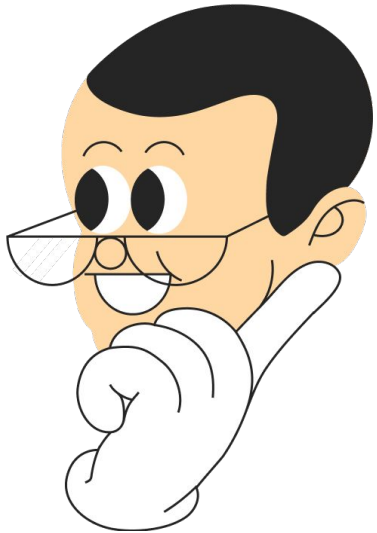
- **json_file = json.load(f)**
  загрузка JSON из файла и сохранение в dict или list
- **json_list = json.loads(json_text)**
  загрузка JSON из строки и сохранение в dict или list



Преобразование Python в JSON

import json

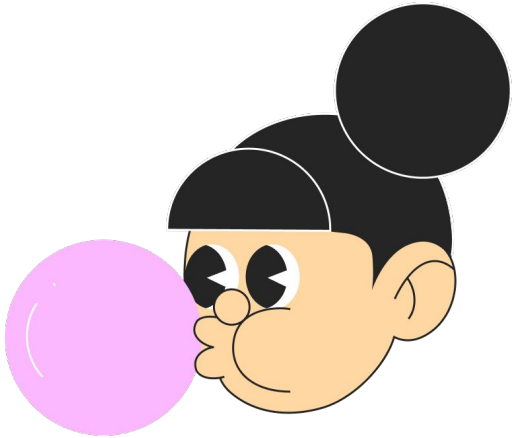
- **json.dump(my_dict, f)**
  сохранение dict или list в файле в виде JSON
- **dict_to_json_text = json.dumps(my_dict)**
  сохранение dict или list в виде JSON строки



Дополнительные параметры dump и dumps

```
res = json.dumps(my_dict, indent=2,
                          separators=(',', ':'),
                          sort_keys=True)
```

- Параметр indent отвечает за форматирование с отступами
- Параметр separators принимает на вход кортеж из двух строковых элементов.
  Первый — символ разделитель элементов.
  Второй — разделитель ключа и значения.
- Параметр sort_keys отвечает за сортировку ключей по алфавиту



Перед вами несколько строк кода.
Какой объект будет получен после его выполнения?

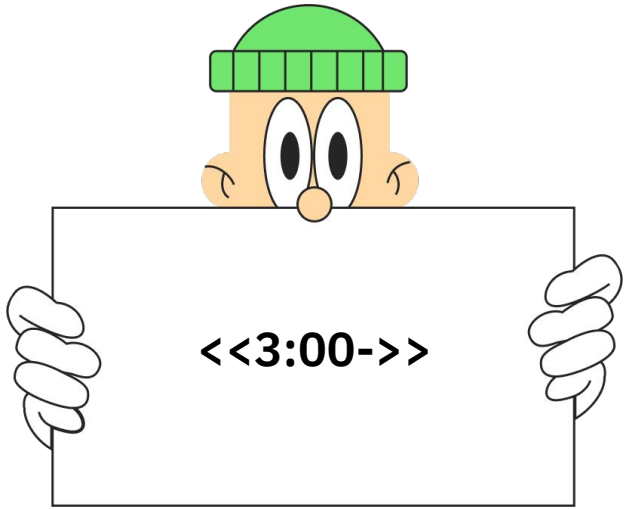
У вас 3 минуты.

JSON

```
import json

a = 'Hello world!'
b = {key: value for key, value in enumerate(a)}

c = json.dumps(b, indent=3, separators=('; ', '= '))
print(c)
```

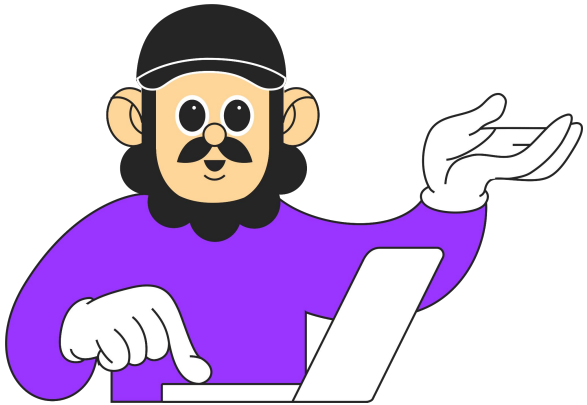


CSV (Comma-Separated Values)

Формат CSV

CSV — текстовый формат, предназначенный для представления табличных данных. Строка таблицы соответствует строке текста, которая содержит одно или несколько полей, разделенных запятыми.

```
"Name","Sex","Age","Height (in)","Weight (lbs)"
"Alex","M",41,74,170
"Bert","M",42,68,166
"Carl","M",32,70,155
```



Чтение CSV

import csv

- **with open('biostats.csv', 'r', newline='') as f:**
  параметр newline='' для работы с CSV
- **csv_file = csv.reader(f)**
  csv_file позволяет построчно читать csv файл в список list

Запись CSV

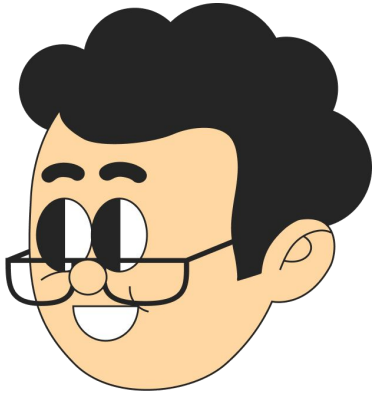
import csv

- **csv_write = csv.writer(f)**
  csv_write позволяет сохранять данные в формате CSV
- **csv_write.writerow(line)**
  сохранение списка в одной строке файла в формате CSV
- **csv_write.writerows(all_data)**
  сохранение матрицы в нескольких строках файла в формате CSV

Чтение CSV в словарь

```
csv_file = csv.DictReader(f, fieldnames=["name", "sex", "age",
"height", "weight", "office"], restkey="new", restval="Main Office")
```

- **fieldnames** — список заголовков столбцов, ключей словаря
- **restkey** — значение ключа для столбцов, которых нет в fieldnames
- **restval** — значение поля для ключей fieldnames, которых нет в файле CSV



Запись CSV из словаря

import csv

- **Параметры класса DictWriter аналогичны параметрам DictReader**
- **csv_write.writeheader()**
  сохранение первой строки с заголовками в порядке их перечисления в параметре fieldnames
- **csv_write.writerow(line)**
  сохранение списка в одной строке файла в формате CSV
- **csv_write.writerows(all_data)**
  сохранение матрицы в нескольких строках файла в формате CSV

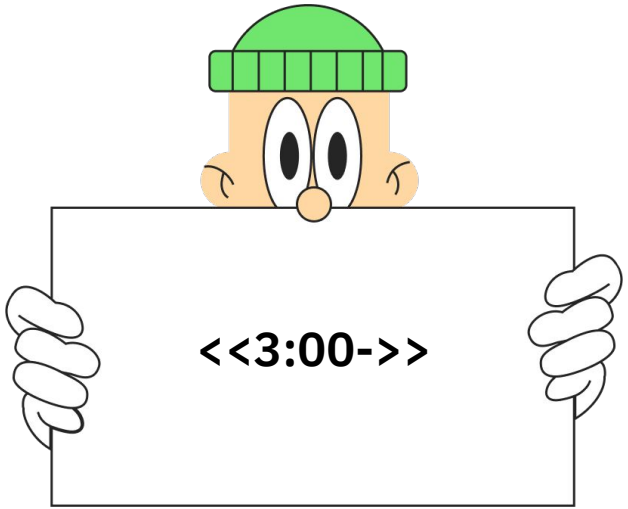
Перед вами несколько строк кода.
Что будет записано в файл после его выполнения?

У вас 3 минуты.

CSV

```
import csv

with open('quest.csv', 'w', newline='',
encoding='utf-8') as f_write:
    csv_write = csv.DictWriter(f_write,
fieldnames=["number", "name", "data"],
restval='Hello world!', dialect='excel',
delimiter='#', quotechar='=',
quoting=csv.QUOTE_NONNUMERIC)
    csv_write.writeheader()
    dict_row = {}
    for i in range(10):
        dict_row['number'] = i
        dict_row['name'] = str(i)
        csv_write.writerow(dict_row)
```



Модуль Pickle

Модуль Pickle

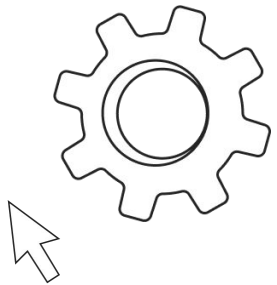
Модуль pickle не занимается проверкой потока байт на безопасность перед распаковкой. Не используйте его с тем набором байт, безопасность которого не можете гарантировать.

```
import pickle

res = pickle.loads(b"cos\nsystem\n(S'echo Hello world!'\ntR.")
print(res)
```

Допустимые типы данных для преобразования

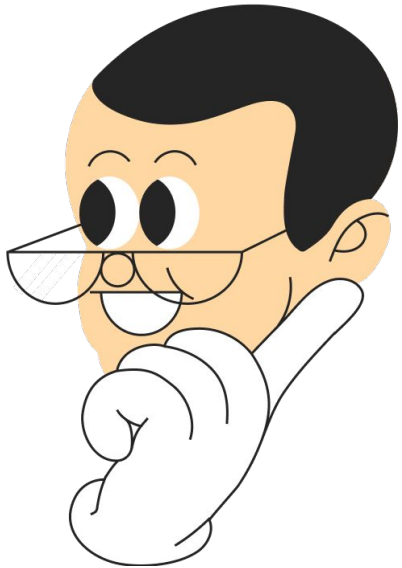
- None, True и False;
- int, float, complex;
- str, bytes, bytearrays;
- tuple, list, set, dict если они содержат объекты, обрабатываемые pickle;
- встроенные функции и функции созданные разработчиком и доступные из верхнего уровня модуля, кроме lambda функций;
- классы доступные из верхнего уровня модуля;
- экземпляры классов, если pickle смог обработать их дандер __dict__ или результат вызова метода __getstate__().



Сериализация

import pickle

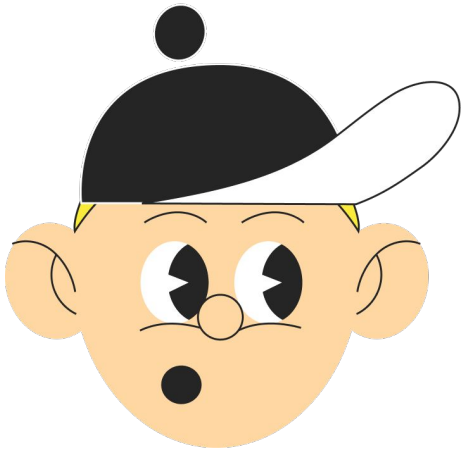
- **pickle.dump(my_dict, f)**
  сохранение объекта в бинарном файле
- **result = pickle.dumps(my_dict)**
  сохранение объекта в строку байт



Десериализация

import pickle

- **new_dict = pickle.load(f)**
  загрузка из бинарного файла и сохранение в объекта
- **new_dict = pickle.loads(data)**
  получение объекта из бинарной строки



Перед вами несколько строк кода.
Что будет записано в файл после его выполнения?

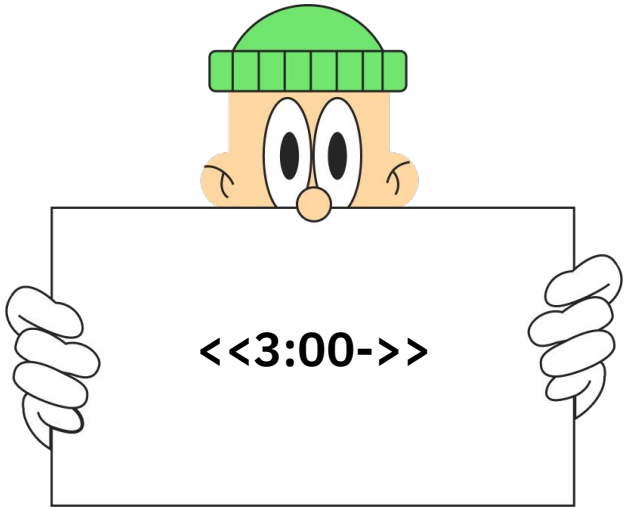
У вас 3 минуты.

Pickle

```
import pickle

my_dict = {'numbers': [42, 4.1415, (7 + 3j)],
           'functions': (sum, max),
           'others': {False, True, 'Hello
world!'}
           }

res = pickle.dumps(my_dict)
with open('quest.pickle', 'wb') as f:
    pickle.dump(f'{res = }', f)
```



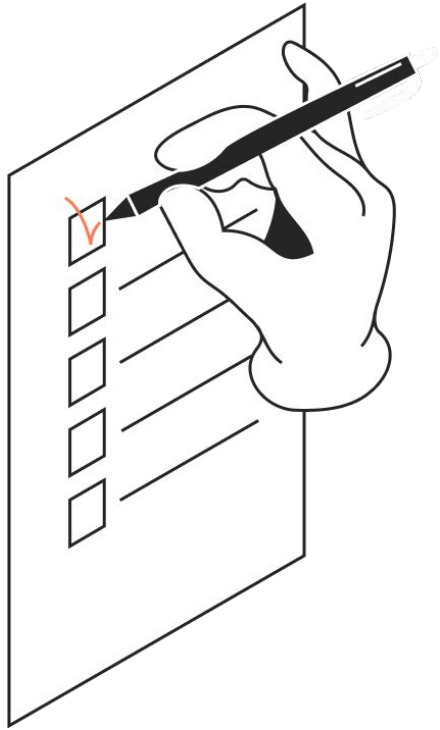
Итоги занятия

На этой лекции мы

- 📌 Разобрались в сериализации и десериализации данных
- 📌 Изучили самый популярный формат сериализации — JSON
- 📌 Узнали о чтении и записи таблиц в формате CSV
- 📌 Разобрались с внутренним сериализатором Python — модулем pickle

Задание

Возьмите код прошлых уроков и попытайтесь сериализовать используемые в нём объекты на основе знаний с сегодняшнего занятия. Попробуйте все варианты сериализации и десериализации в разных форматах.



Спасибо за внимание