

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчет по лабораторной работе №2.16 по дисциплине:  
основы программной инженерии**

Выполнила:

студент группы ПИЖ-б-о-20-1

Лазарева Дарья Олеговна

Проверил:

доцент кафедры инфокоммуникаций

Романкин Р.А.

Ставрополь, 2022 г.

## ВЫПОЛНЕНИЕ:

### 1. Запись списка или словаря в файл:

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      import json
5      my_list = ['foo', 'bar']
6
7      contents = json.dumps(my_list)
8      with open("foo.txt", "w", encoding="utf-8") as f:
9          f.write(contents)
10
11     with open("foo.txt", "w", encoding="utf-8") as f:
12         json.dump(my_list, f)
```

foo.txt – Блокнот

Файл Правка Формат Вид Справка

["foo", "bar"]

### 2. Чтение списка или словаря из файла

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5      import json
6      with open("foo.txt", "r") as f:
7          contents = f.read()
8      my_list = json.loads(contents)
9
10     with open("foo.txt", "r") as f:
11         my_list = json.load(f)
12
```

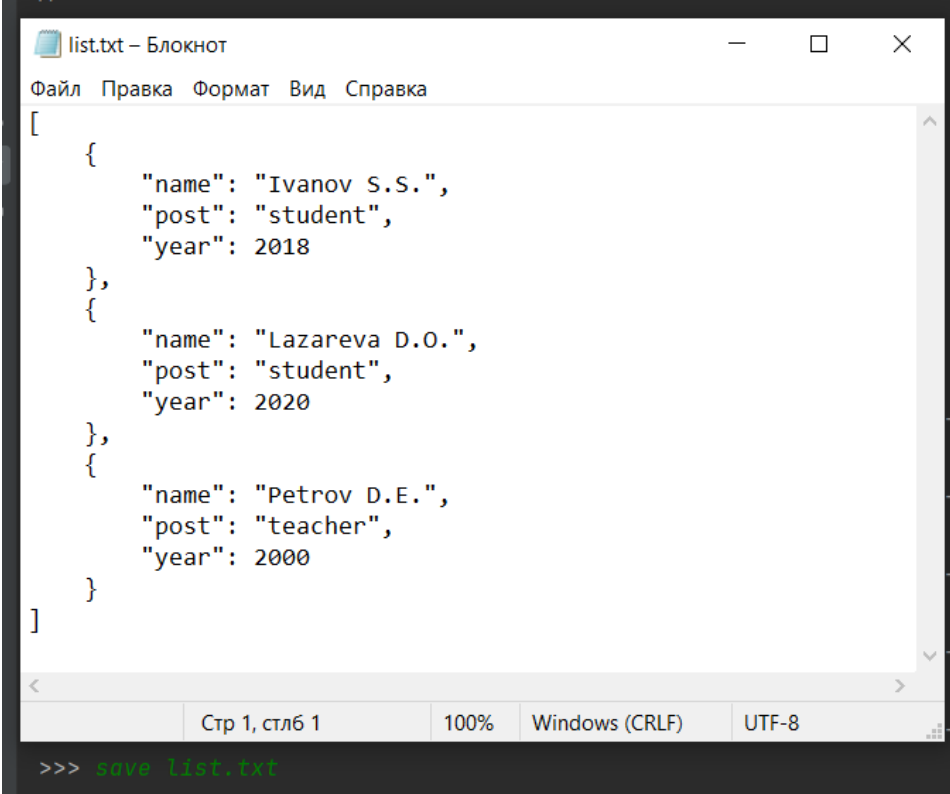
### 3. Результат выполнения 1 примера

```
>>> add
Фамилия и инициалы? Lazareva D.O.
Должность? student
Год поступления? 2020
>>> add
Фамилия и инициалы? Petrov D.E.
Должность? teacher
Год поступления? 2000
>>> add
Фамилия и инициалы? Ivanov S.S.
Должность? student
Год поступления? 2018
>>> list
```

No	Ф.И.О.	Должность	Год
1	Ivanov S.S.	student	2018
2	Lazareva D.O.	student	2020
3	Petrov D.E.	teacher	2000

```
>>>
```

### 4. Выполнение команды list



The screenshot shows a Notepad window titled "list.txt - Блокнот". The menu bar includes "Файл", "Правка", "Формат", "Вид", and "Справка". The text area contains a JSON array with three objects, each representing a person's record. The status bar at the bottom indicates "Стр 1, столб 1", "100%", "Windows (CRLF)", and "UTF-8". Below the status bar, the command ">>> save list.txt" is visible.

```
[
  {
    "name": "Ivanov S.S.",
    "post": "student",
    "year": 2018
  },
  {
    "name": "Lazareva D.O.",
    "post": "student",
    "year": 2020
  },
  {
    "name": "Petrov D.E.",
    "post": "teacher",
    "year": 2000
  }
]
```

```
>>> save list.txt
```

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import json
import sys
from datetime import date

def get_worker():
    surname = input("Фамилия: ")
    name = input("Имя: ")
    number = int(input("Номер телефона: "))
    date_obj = input("Дата рождения: ").split('.')
    return {
        'surname': surname,
        'name': name,
        'number': number,
        'date_obj': date_obj,
    }

def save_workers(file_name, staff):
    with open(file_name, "w", encoding="utf-8") as fout:
        json.dump(staff, fout, ensure_ascii=False, indent=4)

def load_workers(file_name):
    with open(file_name, "r", encoding="utf-8") as fin:
        return json.load(fin)

def main():
    workers = []

    while True:
        command = input(">>> ").lower()

        if command == 'exit':
            break

        elif command == 'add':
            worker = get_worker()

            workers.append(worker)

            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            for num, elem in enumerate(workers):
                print(f"{num+1}.\n{str(elem['surname'])} {str(elem['name'])}\n"
                    f"Номер телефона: {str(elem['number'])}\nДата рождения: {elem['date_obj']}")

        elif command.startswith('select'):
            surname = input("Введите фамилию: ")
            for elem in workers:
                if elem['surname'] == surname:
                    print(f"Имя: {str(elem['name'])}\nНомер телефона: {str(elem['number'])}\n"
                        f"Дата рождения: {elem['date_obj']}")
                    return
            else:
                print("Фамилии не найдено")

            if command == 'exit':
                break

```

```

elif command.startswith("save "):
    parts = command.split(maxsplit=1)
    file_name = parts[1]
    save_workers(file_name, workers)

elif command.startswith("load "):
    parts = command.split(maxsplit=1)
    file_name = parts[1]
    workers = load_workers(file_name)

elif command == 'help':
    print("Список команд:\n")
    print("add - добавить работника;")
    print("list - вывести список работников;")
    print("select <стаж> - запросить работников со стажем;")
    print("help - отобразить справку;")
    print("load - загрузить данные из файла;")
    print("save - сохранить данные в файл;")
    print("exit - завершить работу с программой.")

else:
    print(f"Неизвестная команда {command}", file=sys.stderr)

if __name__ == '__main__':
    main()

```

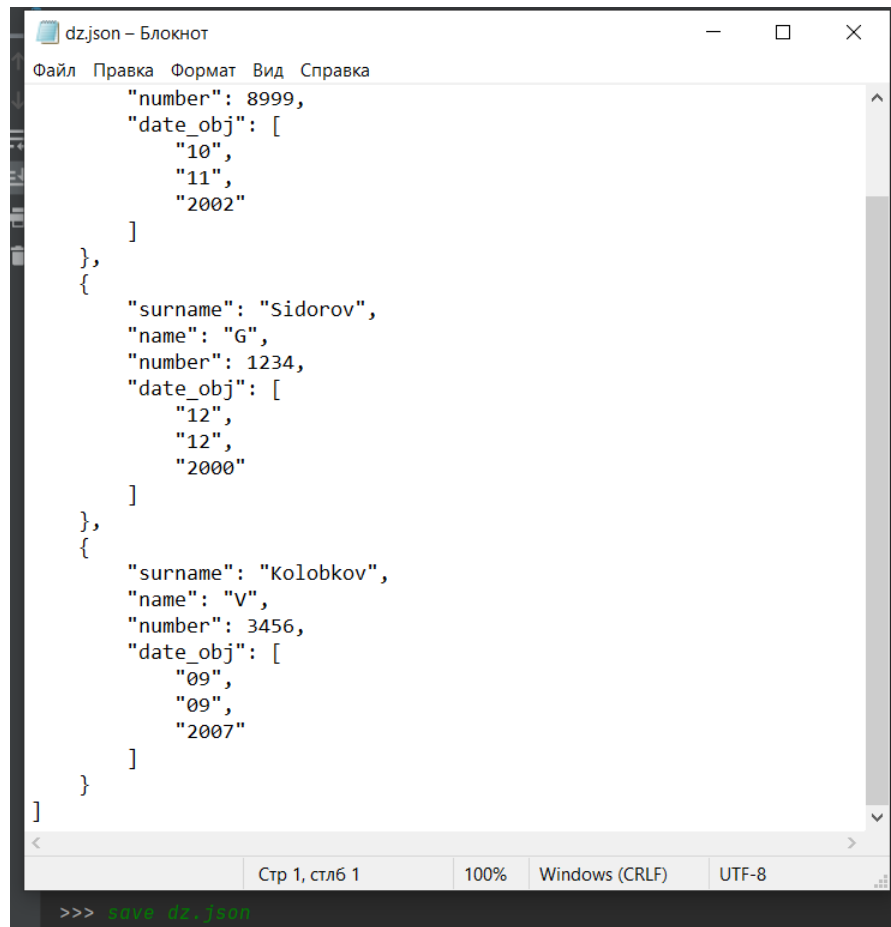
## 5. Выполнение индивидуального примера

```

>>> add
Фамилия: Lazareva
Имя: D
Номер телефона: 8999
Дата рождения: 10.11.2002
>>> add
Фамилия: Sidorov
Имя: G
Номер телефона: 1234
Дата рождения: 12.12.2000
>>> add
Фамилия: Kolobkov
Имя: V
Номер телефона: 3456
Дата рождения: 09.09.2007
>>> list
1.
Lazareva D
Номер телефона: 8999
Дата рождения: ['10', '11', '2002']
2.
Sidorov G
Номер телефона: 1234
Дата рождения: ['12', '12', '2000']
3.
Kolobkov V
Номер телефона: 3456
Дата рождения: ['09', '09', '2007']

```

## 6. Выполнение команды save



```
dz.json - Блокнот
Файл  Правка  Формат  Вид  Справка

{
  "number": 8999,
  "date_obj": [
    "10",
    "11",
    "2002"
  ]
},
{
  "surname": "Sidorov",
  "name": "G",
  "number": 1234,
  "date_obj": [
    "12",
    "12",
    "2000"
  ]
},
{
  "surname": "Kolobkov",
  "name": "V",
  "number": 3456,
  "date_obj": [
    "09",
    "09",
    "2007"
  ]
}
]

Стр 1, столб 1    100%    Windows (CRLF)    UTF-8
>>> save dz.json
```

### Контрольные вопросы:

1. Для чего используется JSON? JSON — это формат, который хранит структурированную информацию и в основном используется для передачи данных между сервером и клиентом. Файл JSON представляет собой более простую и лёгкую альтернативу расширению с аналогичными функциями XML.

2. Какие типы значений используются в JSON?

JSON-текст представляет собой (в закодированном виде) одну из двух структур:

- Набор пар ключ: значение. В различных языках это реализовано как запись, структура, словарь, хеш-таблица, список с ключом или ассоциативный массив. Ключом может быть только строка.
- Упорядоченный набор значений. Во многих языках это реализовано как массив, вектор, список или последовательность.

В качестве значений в JSON могут быть использованы:

- запись — это неупорядоченное множество пар ключ: значение, заключённое в фигурные скобки «{ }»;
- массив (одномерный) — это упорядоченное множество значений. Массив заключается в квадратные скобки «[ ]»;
- число;

- литералы true, false, null;
- строка – упорядоченное множество из нуля или более символов юникода, заключенное в двойные кавычки.

### 3. Как организована работа со сложными данными в JSON?

#### Вложенные объекты

JSON может содержать другие вложенные объекты в JSON, в дополнение к вложенным массивам. Такие объекты и массивы будут передаваться, как значения, назначенные ключам и будут представлять собой связку ключ-значение. Фигурные скобки везде используются для формирования вложенного JSON объекта с ассоциированными именами пользователей и данными локаций для каждого из них. Как и с любым другим значением, используя объекты, двоеточие используется для разделения элементов.

#### Вложенные массивы

Данные также могут быть вложены в формате JSON, используя JavaScript массивы, которые передаются как значения. JavaScript использует квадратные скобки [ ] для формирования массива. Мы можем использовать массив при работе с большим количеством данных, которые могут быть легко сгруппированы вместе, как например, если есть несколько разных сайтов и профайлов в социальных сетях ассоциированных с одним пользователем.

### 4. Самостоятельно ознакомьтесь с форматом данных JSON5? В чем отличие этого формата от формата данных JSON?

JSON5 — предложенное расширение формата json в соответствии с синтаксисом ECMAScript 5, вызванное тем, что json используется не только для общения между программами, но и создаётся/редактируется вручную. Файл JSON5 всегда является корректным кодом ECMAScript 5. JSON5 обратно совместим с JSON. Для некоторых языков программирования уже существуют парсеры json5.

#### Некоторые нововведения:

- Поддерживаются как однострочные //, так и многострочные /\* \*/ комментарии.
- Записи и списки могут иметь запятую после последнего элемента (удобно при копировании элементов).

- Ключи записей могут быть без кавычек, если они являются валидными идентификаторами ECMAScript 5.
- Строки могут заключаться как в одинарные, так и в двойные кавычки.
- Числа могут быть в шестнадцатеричном виде, начинаться или заканчиваться десятичной точкой, включать Infinity, -Infinity, NaN и -NaN, начинаться со знака +.

Проще говоря, он убирает некоторые ограничения JSON, расширяя его синтаксис.

5. Какие средства языка программирования Python могут быть использованы для работы с данными в формате JSON5?

Существует пакет PyJSON5, который содержит множество функций для расширения функционала JSON.

Ниже представлены функции для сериализации данных

## Quick Encoder Summary

<code>encode (data, *[, options])</code>	Serializes a Python object as a JSON5 compatible string.
<code>encode_bytes (data, *[, options])</code>	Serializes a Python object to a JSON5 compatible bytes string.
<code>encode_callback (data, cb[, supply_bytes, ...])</code>	Serializes a Python object into a callback function.
<code>encode_io (data, fp[, supply_bytes, options])</code>	Serializes a Python object into a file-object.
<code>encode_noop (data, *[, options])</code>	Test if the input is serializable.
<code>dump (obj, fp, **kw)</code>	Serializes a Python object to a JSON5 compatible string.
<code>dumps (obj, **kw)</code>	Serializes a Python object to a JSON5 compatible string.
<code>Options</code>	Customizations for the <code>encoder_*(...)</code> function family.
<code>Json5EncoderException</code>	Base class of any exception thrown by the serializer.
<code>Json5UnstringifiableType ((message, ...))</code>	The encoder was not able to stringify the input, or it was too large.

Функции для кодирования/декодирования данных:



## Quick Decoder Summary

<code>decode</code> (data[, maxdepth, some])	Decodes JSON5 serialized data from an <code>str</code> object.
<code>decode_latin1</code> (data[, maxdepth, some])	Decodes JSON5 serialized data from a <code>bytes</code> object.
<code>decode_buffer</code> (obj[, maxdepth, some, wordlength])	Decodes JSON5 serialized data from an object that s
<code>decode_callback</code> (cb[, maxdepth, some, args])	Decodes JSON5 serialized data by invoking a callback
<code>decode_io</code> (fp[, maxdepth, some])	Decodes JSON5 serialized data from a file-like object
<code>load</code> (fp, **kw)	Decodes JSON5 serialized data from a file-like object
<code>loads</code> (s, *[, encoding])	Decodes JSON5 serialized data from a string.
<code>Json5DecoderException</code> ([message, result])	Base class of any exception thrown by the parser.
<code>Json5NestingTooDeep</code>	The maximum nesting level on the input data was ex
<code>Json5EOF</code>	The input ended prematurely.
<code>Json5IllegalCharacter</code> ([message, result, ...])	An unexpected character was encountered.
<code>Json5ExtraData</code> ([message, result, character])	The input contained extraneous data.
<code>Json5IllegalType</code> ([message, result, value])	The user supplied callback function returned illegal d.

6. Какие средства предоставляет язык Python для сериализации данных в формате JSON?

Сериализация данных в формат JSON:

`json.dump()` # конвертировать python объект в json и записать в файл

`json.dumps()` # тоже самое, но в строку

7. В чем отличие функций `json.dump()` и `json.dumps()`?

`json.dumps()` конвертирует python объект в json и записывает его в строку вместо записи в файл.

8. Какие средства предоставляет язык Python для десериализации данных из формата JSON?

Десериализация данных из формата JSON:

`json.load()` # прочитать json из файла и конвертировать в python объект

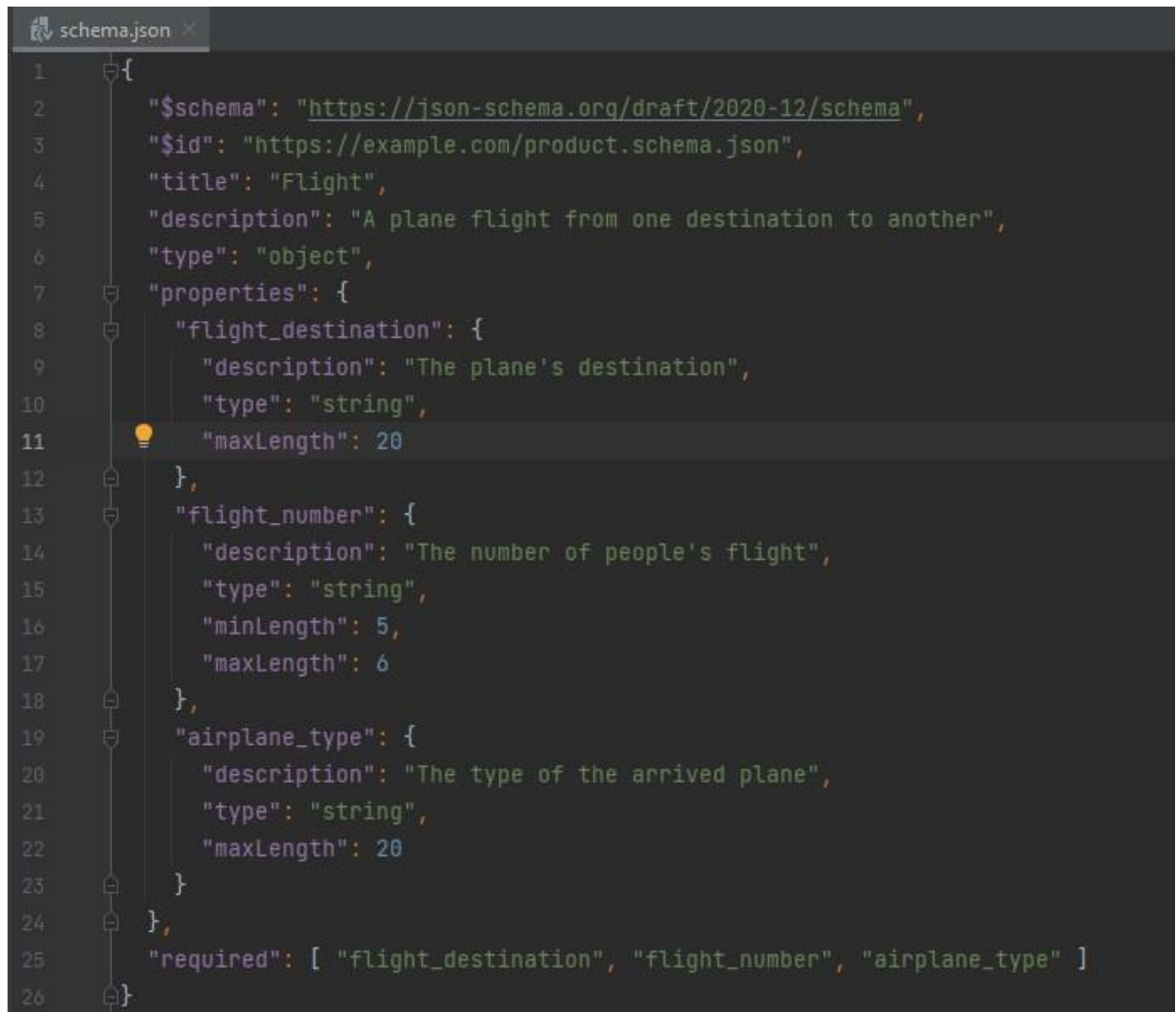
`json.loads()` # тоже самое, но из строки с json (s на конце от string/строка)

9. Какие средства необходимо использовать для работы с данными формата JSON, содержащими кириллицу?

Использование кодировки UTF-8 или `ensure_ascii=False`

10. Самостоятельно ознакомьтесь со спецификацией JSON Schema? Что такое схема данных? Приведите схему данных для примера 1.

Схема данных представляет собой код, который используется для валидации данных в формате JSON. Схема данных:



```
1  {
2    "$schema": "https://json-schema.org/draft/2020-12/schema",
3    "$id": "https://example.com/product.schema.json",
4    "title": "Flight",
5    "description": "A plane flight from one destination to another",
6    "type": "object",
7    "properties": {
8      "flight_destination": {
9        "description": "The plane's destination",
10       "type": "string",
11       "maxLength": 20
12     },
13     "flight_number": {
14       "description": "The number of people's flight",
15       "type": "string",
16       "minLength": 5,
17       "maxLength": 6
18     },
19     "airplane_type": {
20       "description": "The type of the arrived plane",
21       "type": "string",
22       "maxLength": 20
23     }
24   },
25   "required": [ "flight_destination", "flight_number", "airplane_type" ]
26 }
```