

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №11

Работа с данными формата JSON в языке Python.

По дисциплине «Технологии программирования и алгоритмизация»

Выполнил студент группы ИВТ-б-о-20-1

Бобров Н. В. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р. А. _____

(подпись)

Ставрополь 2021

Цель работы: приобретение навыков по работе с данными формата JSON с помощью языка программирования Python версии 3.x.

Ход работы:

Ссылка на репозиторий: <https://github.com/nbobrov8/lab18>

1. Создал общедоступный репозиторий и клонировал его на локальный сервер.
2. Изучил теоретический материал и проработал примеры работы с JSON.

```
77 def save_workers(file_name, staff):
78     """
79     Сохранить всех работников в файле JSON
80     """
81     with open(file_name, "w", encoding="utf-8") as fout:
82         json.dump(staff, fout, ensure_ascii=False, indent=4)
83
84
85 def load_workers(file_name):
86     """
87     Загрузить всех работников из файла JSON
88     """
89     with open(file_name, "r", encoding="utf-8") as fin:
90         return json.load(fin)
91
```

Рисунок 1 – Функции сохранения и загрузки работников

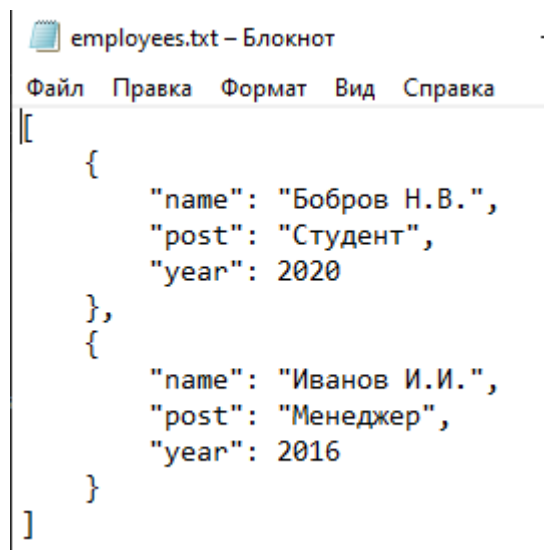
```
131 elif command.startswith("save "):
132     # Разбить команду на части для выделения имени файла.
133     parts = command.split(maxsplit=1)
134     # Получить имя файла.
135     file_name = parts[1]
136     # Сохранить данные в файл с заданным именем.
137     save_workers(file_name, workers)
138
139 elif command.startswith("load "):
140     # Разбить команду на части для выделения имени файла.
141     parts = command.split(maxsplit=1)
142     # Получить имя файла.
143     file_name = parts[1]
144     # Загрузить данные файла с заданным именем.
145     workers = load_workers(file_name)
146
```

Рисунок 2 – Обращение к функциям по командам

3. Сделал проверку работоспособности загрузки и сохранения файлов. Для этого добавил данные в таблицу. С помощью команды save сохранил свою таблицу в формате txt.

```
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          | Должность | Год |
+-----+-----+-----+-----+
|  1 | Бобров Н.В.              | Студент   | 2020 |
+-----+-----+-----+-----+
|  2 | Иванов И.И.              | Менеджер  | 2016 |
+-----+-----+-----+-----+
>>> save employees.txt
```

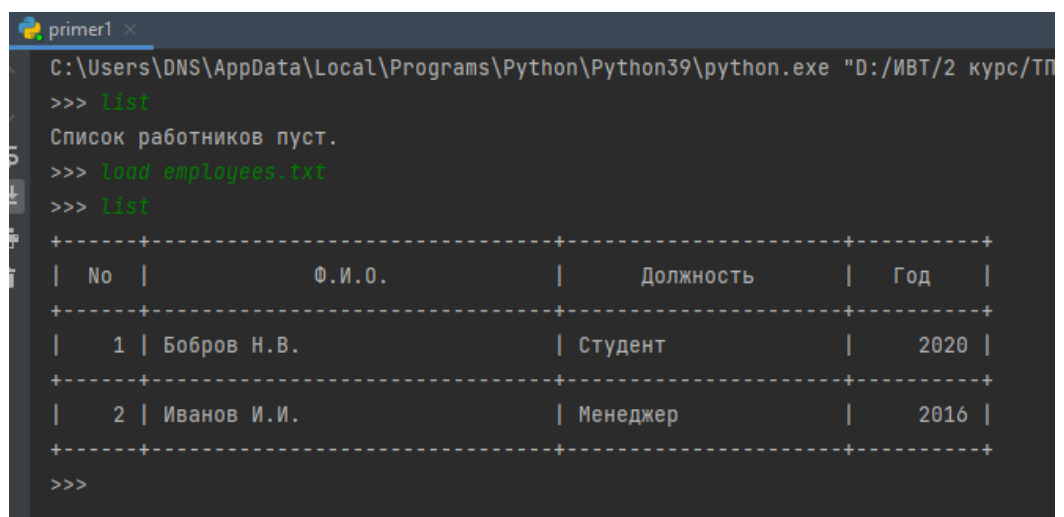
Рисунок 3 – Таблица с данными



```
employees.txt – Блокнот
Файл  Правка  Формат  Вид  Справка
[
  {
    "name": "Бобров Н.В.",
    "post": "Студент",
    "year": 2020
  },
  {
    "name": "Иванов И.И.",
    "post": "Менеджер",
    "year": 2016
  }
]
```

Рисунок 4 – Результат сохранения таблицы

4. Затем завершил свою программу и попытался загрузить таблицу.



```
primer1 x
C:\Users\DNS\AppData\Local\Programs\Python\Python39\python.exe "D:/ИВТ/2 курс/ТП
>>> list
Список работников пуст.
>>> load employees.txt
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          | Должность | Год |
+-----+-----+-----+-----+
|  1 | Бобров Н.В.              | Студент   | 2020 |
+-----+-----+-----+-----+
|  2 | Иванов И.И.              | Менеджер  | 2016 |
+-----+-----+-----+-----+
>>>
```

Рисунок 5 – Результат загрузки таблицы

Индивидуальное задание. Вариант 1.

1. В качестве индивидуального задания взял работу из лабораторной работы 2.8. Добавил две функции отвечающие за сохранение и загрузку файлов.

```
83 def save_students(file_name, students):
84     with open(file_name, "w", encoding="utf-8") as fout:
85         json.dump(students, fout, ensure_ascii=False, indent=4)
86
87
88 def load_students(file_name):
89     """
90     Загрузить всех работников из файла JSON
91     """
92     with open(file_name, "r", encoding="utf-8") as fin:
93         return json.load(fin)
94
```

Рисунок 6 – Объявление функций сохранения и загрузки данных

```
elif command.startswith("save"):
    # Разбить команду на части для выделения имени файла.
    parts = command.split(maxsplit=1)
    # Получить имя файла.
    file_name = parts[1]
    # Сохранить данные в файл с заданным именем.
    save_students(file_name, students)

elif command.startswith("load "):
    # Разбить команду на части для выделения имени файла.
    parts = command.split(maxsplit=1)
    # Получить имя файла.
    file_name = parts[1]
    # Загрузить данные файла с заданным именем.
    students = load_students(file_name)
```

Рисунок 7 – Обращение к командам save и load

2. Сделал проверку кода. Для этого внес исходные данные в таблицу и сохранил её.

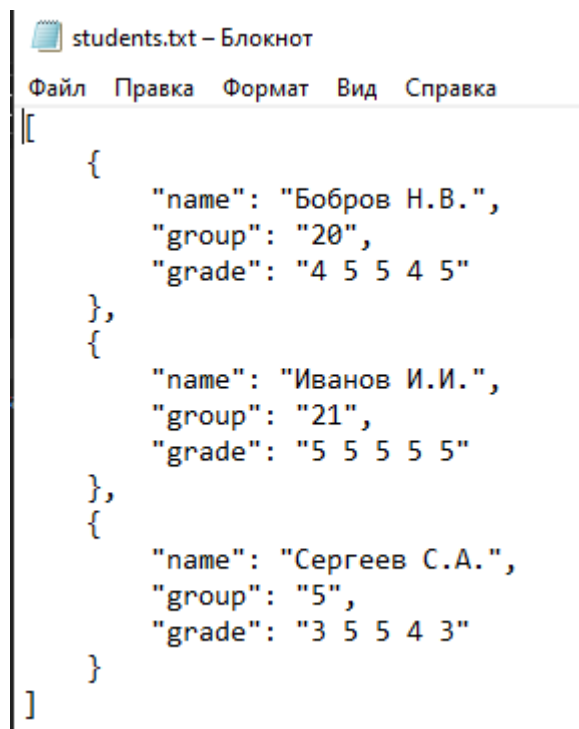
```
>>> list
```

№	Ф.И.О.	Группа	Успеваемость
1	Бобров Н.В.	20	4 5 5 4 5
2	Иванов И.И.	21	5 5 5 5 5
3	Сергеев С.А.	5	3 5 5 4 3

```
>>> save students.txt
```

Рисунок 8 – Сохранение исходной таблицы

3. Проверил наличия файла с расширением txt.



```
students.txt – Блокнот
Файл  Правка  Формат  Вид  Справка
[
  {
    "name": "Бобров Н.В.",
    "group": "20",
    "grade": "4 5 5 4 5"
  },
  {
    "name": "Иванов И.И.",
    "group": "21",
    "grade": "5 5 5 5 5"
  },
  {
    "name": "Сергеев С.А.",
    "group": "5",
    "grade": "3 5 5 4 3"
  }
]
```

Рисунок 9 – Текстовый файл с данными

4. Обновил таблицу и попробовал загрузить текстовый файл с помощью команды load.

```
>>> list
```

№	Ф.И.О.	Группа	Успеваемость
1	Бобров Н.В.	20	4 5 5 4 5
2	Иванов И.И.	21	5 5 5 5 5
3	Сергеев С.А.	5	3 5 5 4 3

```
>>> load students.txt
>>> list
```

№	Ф.И.О.	Группа	Успеваемость
1	Бобров Н.В.	20	4 5 5 4 5
2	Иванов И.И.	21	5 5 5 5 5
3	Сергеев С.А.	5	3 5 5 4 3

Рисунок 10 – Результат загрузки данных

5. Также сделал валидацию данных при загрузке файла с помощью библиотеки jsonschema.

```
89 def load_students(file_name):
90     """
91     Загрузить всех работников из файла JSON
92     """
93     with open(file_name, "r", encoding="utf-8") as fin:
94         file = json.load(fin)
95         print("File loaded")
96         with open("schema.json") as check:
97             schema = json.load(check)
98             validator = jsonschema.Draft7Validator(schema)
99             try:
100                 if not validator.validate(file):
101                     print("Нет ошибок валидации")
102             except jsonschema.exceptions.ValidationError:
103                 print("Ошибка валидации", list(validator.iter_errors(file)))
104                 exit()
105             return file
106
107
```

Рисунок 11 – Валидация данных

6. Имитировал ошибку в данных json, чтобы убедиться в правильности работе валидации.

```
>>> load_students.txt
Ошибка валидации [<ValidationError: "'grade' is a required property">,
```

Рисунок 12 – Вывод ошибки при неправильной загрузке

Контрольные вопросы:

1. Для чего используется JSON?
 - JSON используется для обмена данными, которые являются структурированными и хранятся в файле или в строке кода.
2. Какие типы значений используются в JSON?
 - string;
 - number;
 - object;
 - array;
 - boolean;
 - null.
3. Как организована работа со сложными данными в JSON?

– Данные также могут быть вложены в формате JSON, используя JavaScript массивы, которые передаются как значения. При помощи вложенных массивов и объектов можно создать сложную иерархию данных.

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

Формат обмена данными JSON5 (JSON5) — это надмножество JSON, которое направлено на смягчение некоторых ограничений JSON путем расширения его синтаксиса для включения некоторых продуктов из ECMAScript 5.1.

JSON5 получил следующие новшества:

- строки могут охватывать несколько строк, экранируя новые символы строк;
- числа могут быть шестнадцатеричными;
- допускаются однострочные и многострочные комментарии;
- ключи объектов могут быть без кавычек, если они являются законными идентификаторами ECMAScript;
- объекты и массивы могут заканчиваться запятыми в конце.

Существует одно заметное отличие от JSON: методы `load()` и `loads()` поддерживают выборочную проверку (и отклонение) дубликатов ключей объектов.

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

- `json5.load()`;
- `json5.loads()`;
- `json5.tool()`;
- `json5.dump()`;
- `json5.dumps()`.

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

– Процесс кодирования данных в необходимый формат называется сериализацией. Для того чтобы записать эти данные в файл с форматом JSON в Python, используются функция `dump()` и `dumps()`.

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

– `Dump` отличается от `dumps` тем, что `dump` записывает объект Python в файл JSON, а `dumps` сериализует объект Python и хранит его в виде строки.

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

– Когда есть файл JSON, который необходимо преобразовать в объект Python, тогда проводится десериализация. Для десериализации по аналогии используются две функции: `load()` и `loads()`.

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

– При записи достаточно передать `ensure_ascii=False`, чтобы не экранировать не-ascii символы.

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

Схема JSON – это словарь, который позволяет аннотировать и проверять документы JSON.

Преимущества:

- описывает ваш существующий формат(ы) данных;
- обеспечивает четкую читаемую документацию для человека и машины;
- проверяет данные, которые полезны для автоматизированного тестирования и обеспечения качества предоставляемых клиентом данных.

Пример схемы.

```
Schema = {  
    "type": "object",  
    "employees": {  
        "name": {"type": "string"},
```



```
        "post": {"type": "string"},
        "year": {"type": "string",
                  "format": "date"}
    }
}
```

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе с данными формата JSON с помощью языка программирования Python версии 3.x.