

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2
дисциплины «анализ данных»

Выполнил:
Середа Кирилл Витальевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Работа с данными формата JSON в языке Python

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

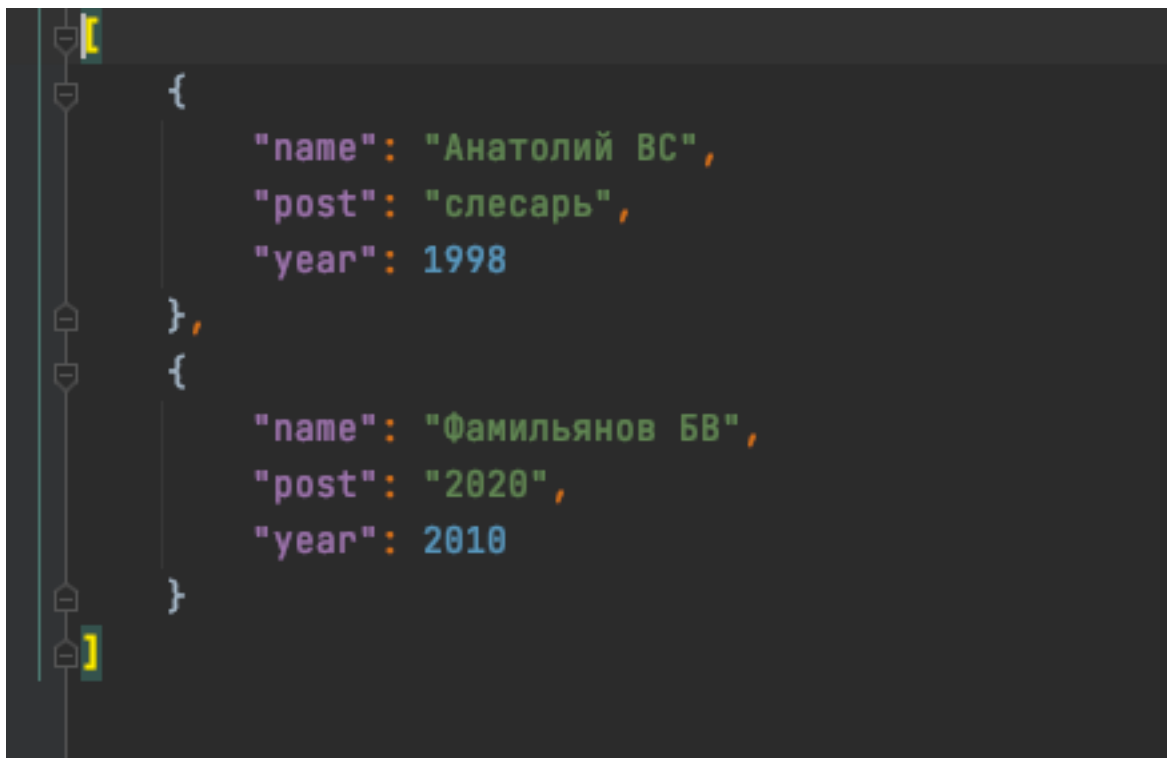
Ход выполнения заданий

1) Создал файл (primer.py), в котором проработал пример лабораторной работы.

```
/Users/MelancholySeal/opt/anaconda3/envs/DA_2/bin/python /Users/MelancholySeal/Documents/GitHub/DA_2/prog/primer.py
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
load - загрузить данные из файла;
save - сохранить данные в файл;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Анатолий ВС
Должность? слесарь
Год поступления? 1998
>>> add
>>> Неизвестная команда фвв
add
Фамилия и инициалы? Фамильянов БВ
Должность? 2020
Год поступления? 2010
>>> list
+-----+-----+-----+
| № |      Ф.И.О.      |      Должность      |      Год      |
+-----+-----+-----+
|  1 | Анатолий ВС      |      слесарь      |      1998      |
|  2 | Фамильянов БВ    |      2020          |      2010      |
+-----+-----+-----+
>>> save data.json
>>>
```

Рисунок 1 – Проработка примера

A screenshot of a code editor showing a JSON array with two objects. The first object has fields 'name' (Анатолий ВС), 'post' (слесарь), and 'year' (1998). The second object has fields 'name' (Фамильянов БВ), 'post' (2020), and 'year' (2010). The code is color-coded: strings in green, numbers in blue, and punctuation in orange. A file explorer on the left shows the file is selected.

```
[
  {
    "name": "Анатолий ВС",
    "post": "слесарь",
    "year": 1998
  },
  {
    "name": "Фамильянов БВ",
    "post": "2020",
    "year": 2010
  }
]
```

Рисунок 2 – Созданный json файл

2) Индивидуальное задание. Переработать программу из лабораторной 2.8 для использования файлов json, а также ввести валидацию данных jsonscheme

```

7 def save_to_json(filepath, data):
8     with open(filepath, "w") as file:
9         json.dump(data, file, ensure_ascii=False, indent=4)
10
11
12 def load_from_json(filename):
13     with open(filename, "r") as file:
14         data = json.load(file)
15     return data
16
17
18 def validate_data(data):
19     try:
20         jsonschema.validate(instance=data, schema=schema)
21         return True
22     except jsonschema.exceptions.ValidationError as e:
23         print(f"Ошибка валидации: {e}")
24         return False
25
26
27 def save_command(students):
28     filename = input("Введите имя файла для сохранения данных: ")
29     save_to_json(filename, students)
30     print(f"Данные успешно сохранены в файл {filename}")
31
32
33 def load_command(students):
34     filename = input("Введите имя файла для загрузки данных: ")
35     loaded_data = load_from_json(filename)
36     if validate_data(loaded_data):
37         students.clear()
38         students.extend(loaded_data)
39         print(f"Данные успешно загружены из файла {filename}")
40     else:
41         print("Загруженные данные не прошли валидацию")

```

Рисунок 3 – Реализация нового функционала

Ответы на вопросы:

1. JSON (JavaScript Object Notation) используется для обмена данными между приложениями. Он часто используется в веб-разработке для передачи данных между клиентом и сервером.
2. В JSON используются следующие типы значений:
 - Строки (string)
 - Числа (number)

- Логические значения (true или false)
- Массивы (array)
- Объекты (object)
- Null

3. Для работы со сложными данными в JSON можно использовать различные методы сериализации и десериализации, а также обращаться к вложенным объектам и массивам с помощью индексов и ключей.

4. Формат данных JSON5 - это расширение формата JSON, которое добавляет некоторые удобные функции, такие как поддержка комментариев, одинарные кавычки для строковых значений и возможность использования без кавычек для ключей объекта. Основное отличие от формата JSON заключается в дополнительных функциях, которые облегчают чтение и написание JSON-данных.

5. Для работы с данными в формате JSON5 на языке Python вы можете использовать сторонние библиотеки, такие как **json5**, которая предоставляет возможность сериализации и десериализации данных из и в формат JSON5.

6. В языке Python для сериализации данных в формате JSON вы можете использовать модуль **json**. Например, функция **json.dump()** используется для записи данных JSON в файл, а функция **json.dumps()** для преобразования данных JSON в строку.

7. Отличие между **json.dump()** и **json.dumps()** состоит в том, что **json.dump()** записывает данные JSON в файл, в то время как **json.dumps()** возвращает строку JSON.

8. Для десериализации данных из формата JSON в языке Python используется метод **json.load()** для чтения данных из файла или **json.loads()** для чтения данных из строки JSON.

9. Для работы с данными формата JSON, содержащими кириллицу, необходимо убедиться, что файлы сохранены в кодировке UTF-8, а также

использовать корректные настройки при чтении и записи данных с помощью библиотеки **json**.

10. Спецификация JSON Schema - это спецификация, описывающая структуру и ограничения данных в формате JSON. Схема данных определяет типы данных, допустимые значения и другие ограничения.

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