

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 №2.16

Дисциплина: «Анализ данных»
Тема: «Работа с данными формата JSON в языке Python»

Выполнил:
Кенесбаев Хилол Куат улы
2 курс, группа ИБТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных систем
», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Роман Александрович

(подпись)

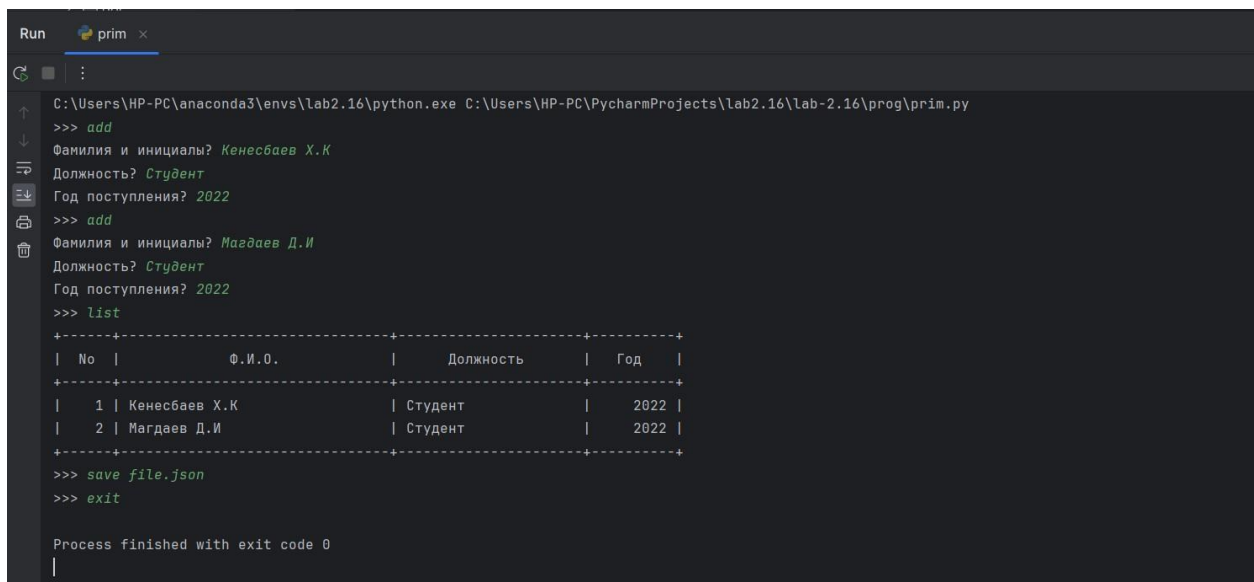
Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

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

Порядок выполнения работы:

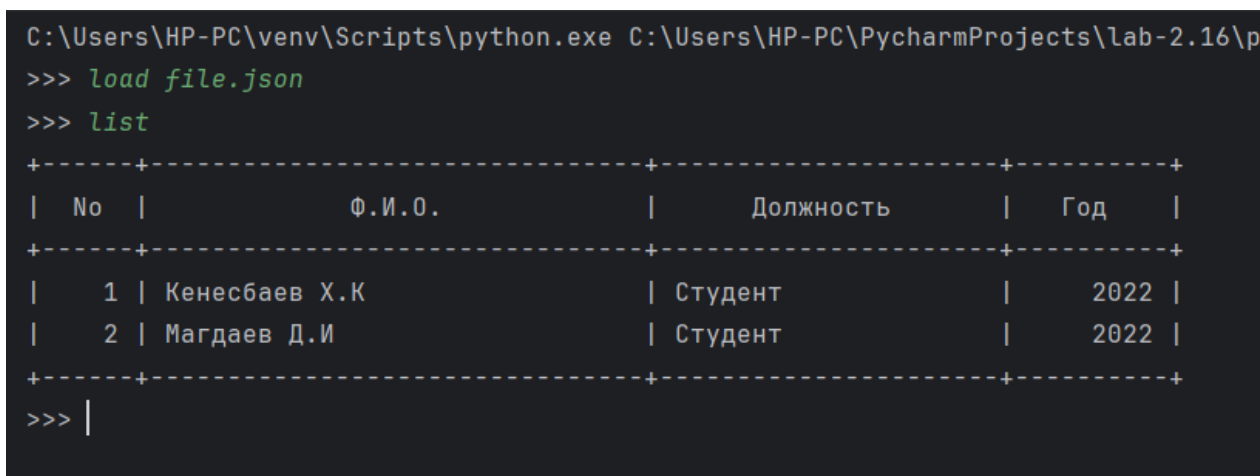
1. Создал новый репозиторий, и клонировал его.
2. Проработал пример лабораторной работы



```
Run prim x
C:\Users\HP-PC\anaconda3\envs\lab2.16\python.exe C:\Users\HP-PC\PycharmProjects\lab2.16\lab-2.16\prog\prim.py
>>> add
Фамилия и инициалы? Кенесбаев Х.К
Должность? Студент
Год поступления? 2022
>>> add
Фамилия и инициалы? Магдаев Д.И
Должность? Студент
Год поступления? 2022
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          | Должность | Год |
+-----+-----+-----+-----+
|  1 | Кенесбаев Х.К          | Студент   | 2022 |
|  2 | Магдаев Д.И            | Студент   | 2022 |
+-----+-----+-----+-----+
>>> save file.json
>>> exit

Process finished with exit code 0
```

Рисунок 1. Сохранение файла .json



```
C:\Users\HP-PC\venv\Scripts\python.exe C:\Users\HP-PC\PycharmProjects\lab-2.16\p
>>> load file.json
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          | Должность | Год |
+-----+-----+-----+-----+
|  1 | Кенесбаев Х.К          | Студент   | 2022 |
|  2 | Магдаев Д.И            | Студент   | 2022 |
+-----+-----+-----+-----+
>>> |
```

Рисунок 2. Загрузка данных файла file.json

Выполнил индивидуальное задание :

Для своего варианта лабораторной работы 2.8 необходимо дополнительно реализовать сохранение и чтение данных из файла формата JSON. Необходимо также проследить за тем, чтобы файлы генерируемый этой программой не попадали в репозиторий лабораторной работы.

```
C:\Users\HP-PC\AppData\Local\Programs\Python\Python39\python.exe C:\Users\HP-PC\PycharmProjects\lab-2.16\prog\individualnoe_1.py
Введите команду: add
Фамилия: Кенесбаев
Имя: Хилол
Знак Зодиака: Козерог
Дата рождения (через пробел дд мм гggg): 28 12 2002
Данные добавлены и упорядочены по датам рождения.
Введите команду: add
Фамилия: Курбаниязов
Имя: Тахир
Знак Зодиака: Рыбы
Дата рождения (через пробел дд мм гggg): 23 03 2002
Данные добавлены и упорядочены по датам рождения.
Введите команду: list
Введите знак Зодиака для поиска: Рыбы
Люди с знаком Зодиака 'Рыбы':
Курбаниязов Тахир - 23.3.2002
Введите команду: save
Введите имя файла для сохранения: ind.json
Данные сохранены в файл ind.json.
Введите команду: load
Введите имя файла для загрузки: ind.json
Введите команду: list
Введите знак Зодиака для поиска: Рыбы
Люди с знаком Зодиака 'Рыбы':
Курбаниязов Тахир - 23.3.2002
Введите команду: |
```

Рисунок 3. Результат работы программы

```
import json

def add_person_to_list(people_list):
    person = {}
    person["фамилия"] = input("Фамилия: ")
    person["имя"] = input("Имя: ")
    person["знак Зодиака"] = input("Знак Зодиака: ")
    birthday = list(map(int, input("Дата рождения (через пробел дд мм гggg): ").split()))
    person["дата рождения"] = birthday
    people_list.append(person)
    people_list.sort(key=lambda x: tuple(x["дата рождения"]))
    print("Данные добавлены и упорядочены по датам рождения.")

def display_people_by_zodiac_sign(people_list, search_zodiac_sign):
    found = False
    print(f"Люди с знаком Зодиака '{search_zodiac_sign}':")
    for person in people_list:
        if person["знак Зодиака"] == search_zodiac_sign:
            print(f"{person['фамилия']} {person['имя']} - {person['дата рождения'][0]}.{person['дата рождения'][1]}.{person['дата рождения'][2]}")
            found = True
    if not found:
        print(f"Нет людей с знаком Зодиака '{search_zodiac_sign}'.")

def save_to_json(people_list, filename):
```

```
with open(filename, "w") as f:
    json.dump(people_list, f, indent=4, ensure_ascii=False)
print(f"Данные сохранены в файл {filename}.")

def load_from_json(filename):
    with open(filename, "r") as f:
        people_list = json.load(f)
    return people_list

if __name__ == "__main__":
    people_data = []

    while True:
        command = input("Введите команду: ").lower()
        if command == "add":
            add_person_to_list(people_data)
        elif command == "list":
            searched_zodiac_sign = input("Введите знак Зодиака для поиска: ")
            display_people_by_zodiac_sign(people_data, searched_zodiac_sign)
        elif command == "save":
            filename = input("Введите имя файла для сохранения: ")
            save_to_json(people_data, filename)
        elif command == "load":
            filename = input("Введите имя файла для загрузки: ")
            people_data = load_from_json(filename)
        elif command == "exit":
            break
        else:
            print("Неизвестная команда.")
```

Ответы на контрольные вопросы:

1. Для чего используется JSON?

JSON (англ. JavaScript Object Notation, обычно произносится как /'dʒeɪsən/ JAY-sən) – текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. Формат JSON был разработан Дугласом Крокфордом.

Несмотря на происхождение от JavaScript (точнее, от подмножества языка стандарта ECMA-262 1999 года), формат считается независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON.

За счёт своей лаконичности по сравнению с XML формат JSON может быть более подходящим для сериализации сложных структур. Применяется в веб-приложениях как для обмена данными между браузером и сервером (AJAX), так и между серверами (программные HTTP-сопряжения).

Легкочитаемый и компактный, JSON представляет собой хорошую альтернативу XML и требует куда меньше форматирования контента.

2. Какие типы значений используются в JSON?

В качестве значений в JSON могут быть использованы:

запись — это неупорядоченное множество пар ключ: значение, заключённое в фигурные скобки «{ }». Ключ описывается строкой, между ним

и значением стоит символ «:». Пары ключ-значение отделяются друг от друга запятыми.

массив (одномерный) — это упорядоченное множество значений. Массив заключается в квадратные скобки «[]». Значения разделяются запятыми. Массив может быть пустым, т.е. не содержать ни одного значения. Значения в пределах одного массива могут иметь разный тип.

число (целое или вещественное).

литералы `true` (логическое значение «истина»), `false` (логическое значение «ложь») и `null`.

строка — это упорядоченное множество из нуля или более символов юникода, заключённое в двойные кавычки. Символы могут быть указаны с использованием escape-последовательностей, начинающихся с обратной косой черты «\» (поддерживаются варианты `'`, `"`, `\`, `\\`, `\t`, `\n`, `\r`, `\f` и `\b`), или записаны шестнадцатеричным кодом в кодировке Unicode в виде `\uFFFF`.

3. Как организована работа со сложными данными в JSON?

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.

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

Формат данных JSON5 является расширенной версией формата JSON, который добавляет несколько улучшений для удобства пользователя. Основные отличия включают поддержку комментариев, однострочные и многострочные, а также разрешенные одинарные кавычки для строковых значений. JSON5 также позволяет использовать запятые в конце списков и свойств, что облегчает редактирование и управление данными.

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

Для работы с JSON5 в Python можно использовать стороннюю библиотеку json5.

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

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

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

Обе эти функции принимают следующие необязательные аргументы: Если `skipkeys = True`, то ключи словаря не базового типа (`str`, `int`, `float`, `bool`, `None`) будут проигнорированы, вместо того, чтобы вызывать исключение `TypeError`.

Если `ensure_ascii = True`, все не-ASCII символы в выводе будут экранированы последовательностями `\uXXXX`, и результатом будет строка, содержащая только ASCII символы. Если `ensure_ascii = False`, строки запишутся как есть.

Если `check_circular = False`, то проверка циклических ссылок будет пропущена, а такие ссылки будут вызывать `OverflowError`.

Если `allow_nan = False`, при попытке сериализовать значение с запятой, выходящее за допустимые пределы, будет вызываться `ValueError` (`nan`, `inf`, `-inf`) в строгом соответствии со спецификацией JSON, вместо того чтобы использовать эквиваленты из JavaScript (`NaN`, `Infinity`, `-Infinity`).

Если `indent` является неотрицательным числом, то массивы и объекты в JSON будут выводиться с этим уровнем отступа. Если уровень отступа 0, отрицательный или `""`, то вместо этого будут просто использоваться новые строки. Значение по умолчанию `None` отражает наиболее компактное представление. Если `indent` - строка, то она и будет использоваться в качестве отступа.

Если `sort_keys = True`, то ключи выводимого словаря будут отсортированы.

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

- конвертировать python объект в json и записать в файл.

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

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

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

`json.loads()` # тоже самое, но из строки с json.

Обе эти функции принимают следующие аргументы:

`object_hook` - опциональная функция, которая применяется к результату декодирования объекта (dict). Используется будет значение, возвращаемое этой функцией, а не полученный словарь.

`object_pairs_hook` - опциональная функция, которая применяется к результату декодирования объекта с определённой последовательностью пар ключ/значение. Будет использован результат, возвращаемый функцией, вместо исходного словаря. Если задан так же `object_hook`, то приоритет отдаётся `object_pairs_hook`.

`parse_float`, если определён, будет вызван для каждого значения JSON с плавающей точкой. По умолчанию, это эквивалентно `float(num_str)`.

`parse_int`, если определён, будет вызван для строки JSON с числовым значением. По умолчанию эквивалентно `int(num_str)`.

`parse_constant`, если определён, будет вызван для следующих строк: "-Infinity", "Infinity", "NaN". Может быть использовано для возбуждения исключений при обнаружении ошибочных чисел JSON.

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

`ensure_ascii=False`.

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

JSON Schema - это спецификация, позволяющая определять формат JSON данных, их структуру, типы данных, ограничения и правила валидации. С помощью JSON Schema можно создавать схемы данных для JSON, что облегчает валидацию и документацию формата JSON.


```
{
  "type":
  "array",
  "items": {
    "type":
    "object",
    "properties":
    {
      "name": {"type": "string"},
      "post": {"type":
        "string"}, "year": {
        "type": "integer",
        "minimum": 2000,
        "maximum": 2024
      }
    },
    "required": ["name", "post", "year"]
  }
}
```

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