Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития Кафедра инфокоммуникаций

ОТЧЁТ

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 ИВТ-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

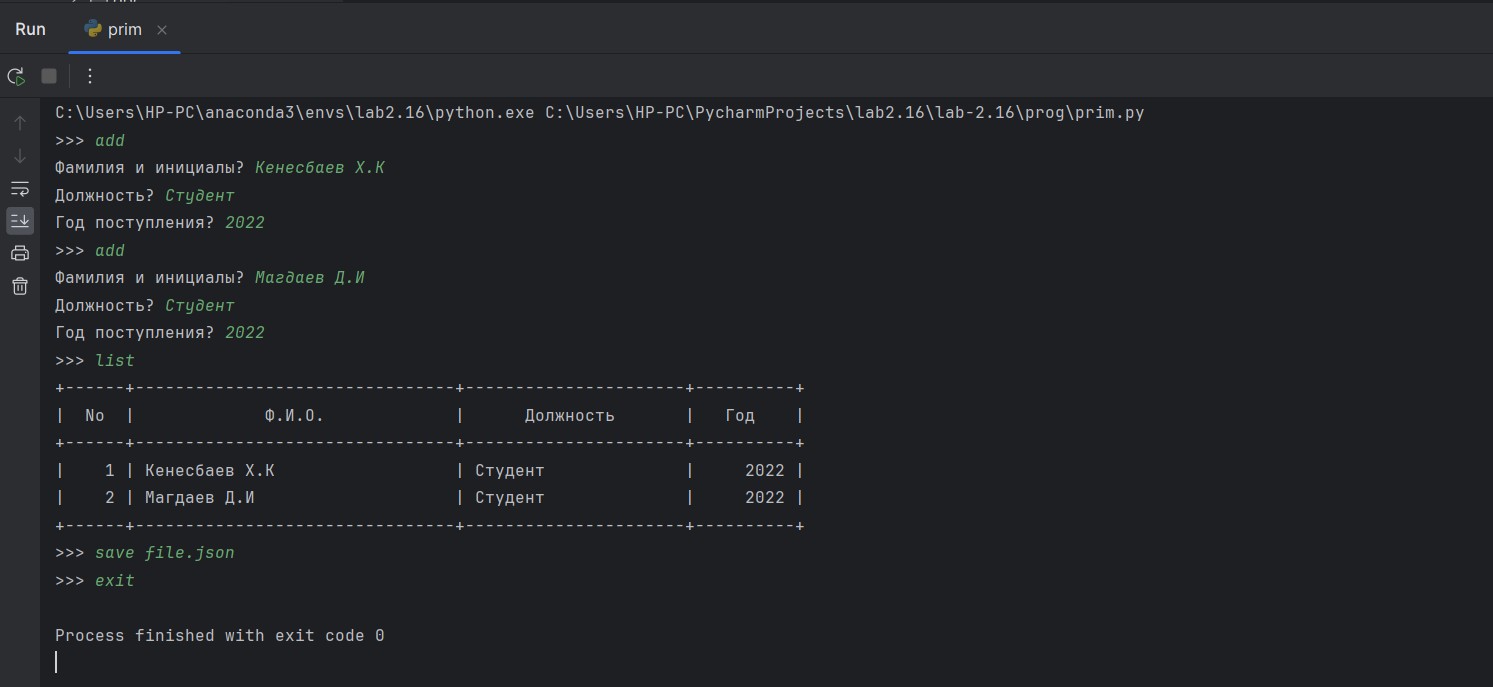


Рисунок 1. Сохранение файла .json

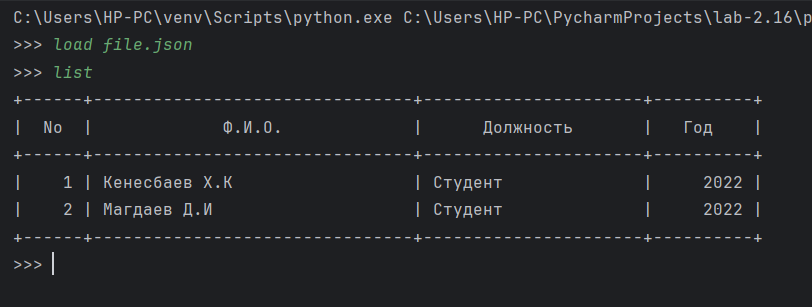


Рисунок 2. Загрузка данных файла file.json

**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.

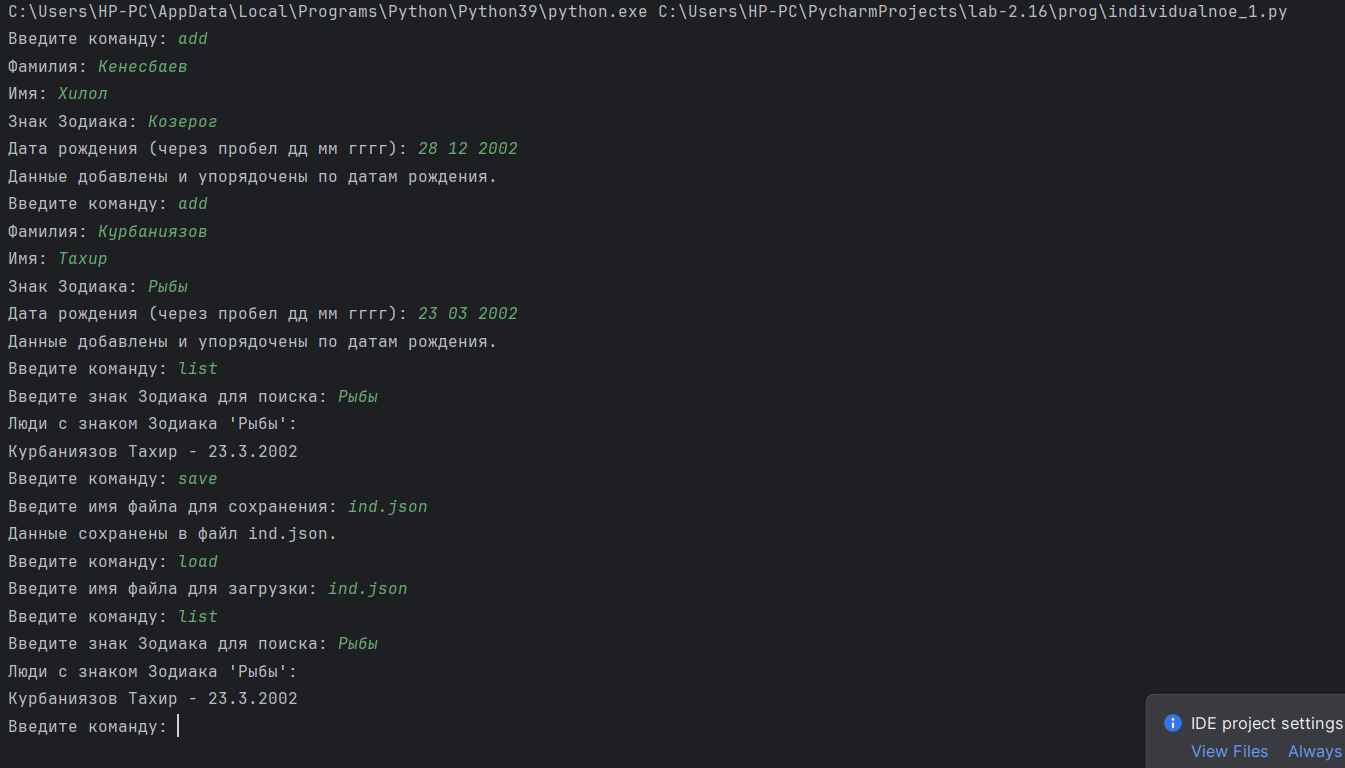


Рисунок 3. Результат работы программы

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 гггг): ").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

/ˈdʒeɪsən/ 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.

* 1. 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.

* 1. 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.

* 1. 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.

* 1. 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.

* 1. 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.

* 1. 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.

* 1. 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). Использоваться 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.

* 1. Какие средства необходимо использовать для работы с данными формата JSON, содержащими кирилицу?

ensure\_ascii=False.

* 1. 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.