

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 ПРАКТИЧЕСКОЙ РАБОТЕ №2
дисциплины «Анализ данных»

Выполнил:

Лейс Алексей Вячеславович
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»

(подпись)

Руководитель практики: кандидат тех.
наук доцент кафедры
инфокоммуникаций: Воронкин Р.А

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Работа с данными формата JSON в языке Python

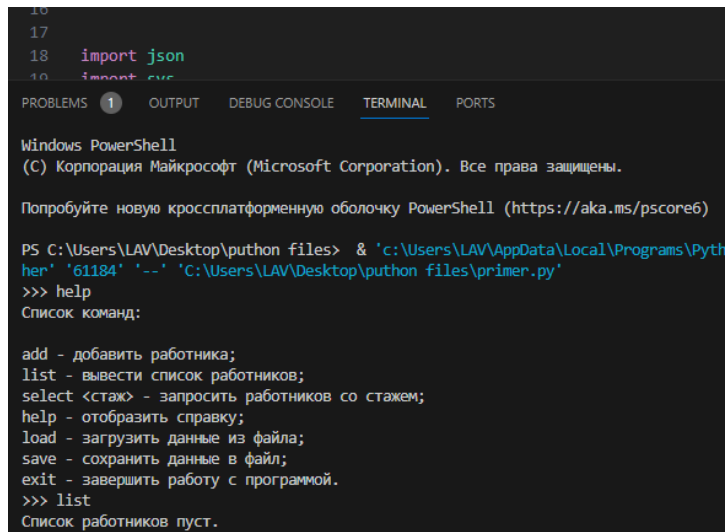
Цель работы: приобретение навыков по работе с данными формата JSON с помощью языка программирования Python версии 3.x.

Порядок выполнения работы:

Для начала устанавливаем виртуальное окружение

`python -m venv .venv`

Пример 1 для примера 1 из лабораторной работы 2.8 добавьте возможность сохранения списка в файл формата JSON и чтения данных из файла JSON.



```
16
17
18 import json
19 import sys

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/powershell)

PS C:\Users\LAV\Desktop\python files> & 'c:\Users\LAV\AppData\Local\Programs\Python\Python38\python.exe' 'C:\Users\LAV\Desktop\python files\primer.py'
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стак> - запросить работников со стажем;
help - отобразить справку;
load - загрузить данные из файла;
save - сохранить данные в файл;
exit - завершить работу с программой.
>>> list
Список работников пуст.
```

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 генерируемые этой программой не попадали в репозиторий лабораторной работы.

```
Task.py 1 Task0.py M X {} data.json primer.py 1
python files > Task0.py > main
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  import json
5
6  def main():
7      # Ввод данных с клавиатуры в список, состоящий из словарей
8      data = []
9      n = int(input("Введите количество записей: "))
10     for i in range(n):
11         record = {}
12         record['расчетный счет плательщика'] = input("Введите расчетный счет плательщика: ")
13         record['расчетный счет получателя'] = input("Введите расчетный счет получателя: ")
14         record['перечисляемая сумма в руб'] = float(input("Введите перечисляемую сумму в рублях: "))
15         data.append(record)
16
17     # Сортировка записей в списке по расчетным счетам плательщиков
18     data = sorted(data, key=lambda x: x['расчетный счет плательщика'])
19
20     # Запись данных в файл формата JSON
21     with open('data.json', 'w') as f:
22         json.dump(data, f, ensure_ascii=False, indent=4)
23
24     # Чтение данных из файла формата JSON
25     with open('data.json') as f:
26         data = json.load(f)
27
28     # Вывод информации о сумме, снятой с расчетного счета плательщика
29     search_account = input("Введите расчетный счет плательщика для поиска: ")
30     found = False
31     for record in data:
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Введите расчетный счет получателя: 45
Введите перечисляемую сумму в рублях: 123
Введите расчетный счет плательщика: 45
Введите расчетный счет получателя: 77
Введите перечисляемую сумму в рублях: 100
Введите расчетный счет плательщика для поиска: 45
{
  "расчетный счет плательщика": "45",
  "расчетный счет получателя": "77",
  "перечисляемая сумма в руб": 100.0
}
PS I:\МВТ\Анализ данных\2\Andate_2>
```

Задание повышенной сложности: Очевидно, что программа в примере 1 и в индивидуальном задании никак не проверяет правильность загружаемых данных формата JSON. В следствие чего, необходимо после загрузки из файла JSON выполнять валидацию загруженных данных. Валидацию данных необходимо производить с использованием спецификации JSON Schema, описанной на сайте <https://json-schema.org/>. Одним из возможных вариантов работы с JSON Schema является использование пакета `jsonschema`, который не является частью стандартной библиотеки Python. Таким образом, необходимо реализовать валидацию загруженных данных с помощью спецификации JSON Schema.

```

import json
from jsonschema import validate

# Загружаем JSON Schema из файла
with open('schema.json', 'r') as schema_file:
    schema = json.load(schema_file)

def load_workers(file_name):
    with open(file_name, "r", encoding="utf-8") as fin:
        data = json.load(fin)
        # Валидация данных из файла с использованием JSON Schema
        validate(instance=data, schema=schema)
    return data

```

```

pythonfiles > {} schema.json > ...
1  {
2      "$schema": "http://json-schema.org/schema#",
3      "type": "object",
4      "properties": {
5          "name": {"type": "string"},
6          "post": {"type": "string"},
7          "year": {"type": "integer"}
8      },
9      "required": ["name", "post", "year"]
10 }
11

```

data.json – Блокнот

Файл Правка Формат Вид Справка

```

[
  {
    "расчетный счет плательщика": "45",
    "расчетный счет получателя": "77",
    "перечисляемая сумма в руб": 100.0
  },
  {
    "расчетный счет плательщика": "77",
    "расчетный счет получателя": "45",
    "перечисляемая сумма в руб": 123.0
  }
]

```

Ответы на вопросы:

1. Для чего используется JSON?
 - JSON используется для обмена данными между приложениями, веб-серверами и клиентами в удобном и легко читаемом формате.
2. Какие типы значений используются в JSON?
 - JSON поддерживает строки, числа, логические значения, массивы, объекты, а также значения null.

3. Как организована работа со сложными данными в JSON?

- Для работы со сложными данными в JSON используются массивы и вложенные объекты, позволяющие организовать и структурировать информацию.

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

- JSON5 расширяет синтаксис JSON, позволяя использовать комментарии, без кавычек для ключей, разрывать строки и использовать дополнительные данные типов.

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

- Для работы с JSON5 в Python могут использоваться библиотеки, например, "json5" или "json5-parser".

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

- Python предоставляет модуль json для сериализации данных в JSON формат.

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

- json.dump() записывает JSON данные в файл, а json.dumps() возвращает JSON в виде строки.

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

- Для десериализации данных из JSON формата в Python используется метод json.loads().

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

- Для работы с данными формата JSON, содержащими кириллицу, необходимо убедиться, что данные корректно кодируются, используя например UTF-8.

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

- Схема данных используется для определения структуры и правил валидации данных в формате JSON. Путем описания свойств объектов JSON, определяется корректность данных в соответствии с требованиями схемы.

- Json для примера 1:

```
{  
  "type": "object",  
  "properties": {  
    "name": { "type": "string" },  
    "post": { "type": "string" },  
    "year": { "type": "number" }  
  },  
  "required": ["name", "post", "year"]  
}
```

Вывод: на основе выполненной работы приобрёл навыки по работе с данными формата JSON с помощью языка программирования Python версии 3.x.