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:
Горбунов Данила Евгеньевич
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 Р А., канд. технических наук, доцент кафедры инфокоммуникаций

_____
(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.

## Ход работы

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python. Выполнил клонирование созданного репозитория.

2. Дополнил файл .gitignore необходимыми правилами.

3. Организовал созданный репозиторий в соответствие с необходимыми требованиями.

4. Проработал примеры лабораторной работы. Создал для них отдельные модули языка Python. Привел в отчете скриншоты результата выполнения программ примеров при различных исходных данных, вводимых с клавиатуры.

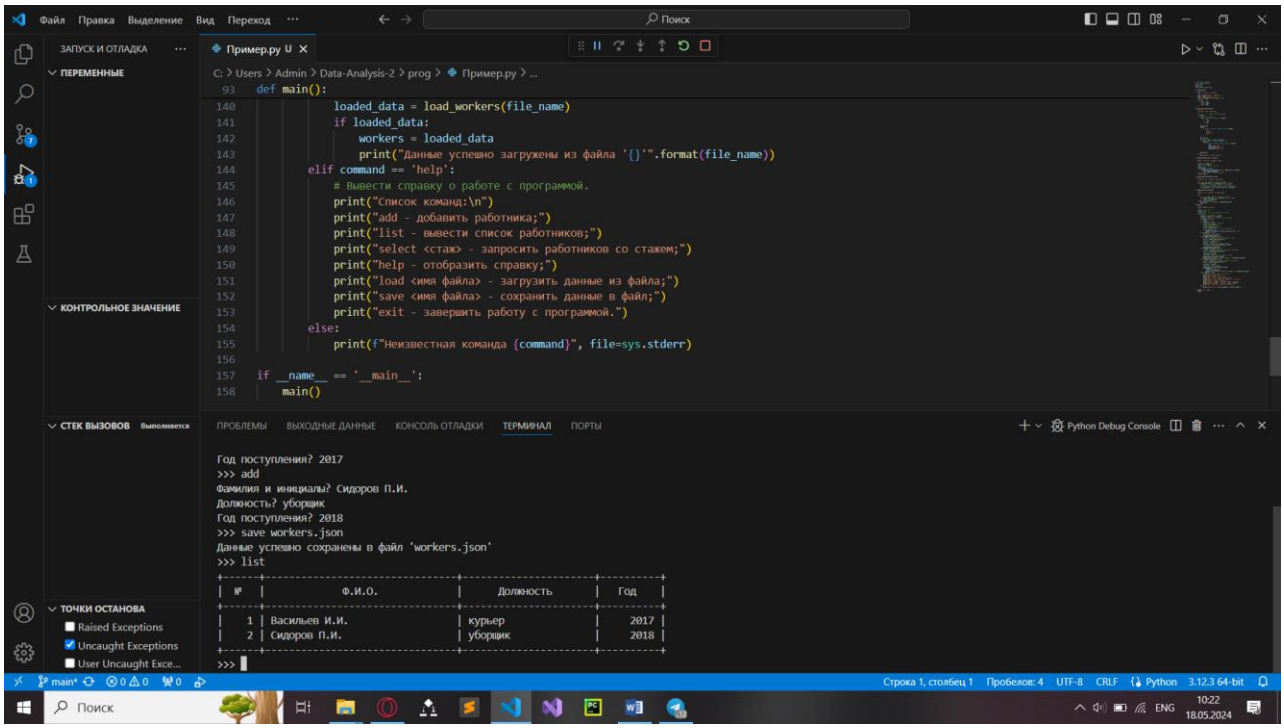


Рисунок 1. Результат работы программы из примера 1

5. Выполнил индивидуальные задания. Привёл в отчете скриншоты работы программ.

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.

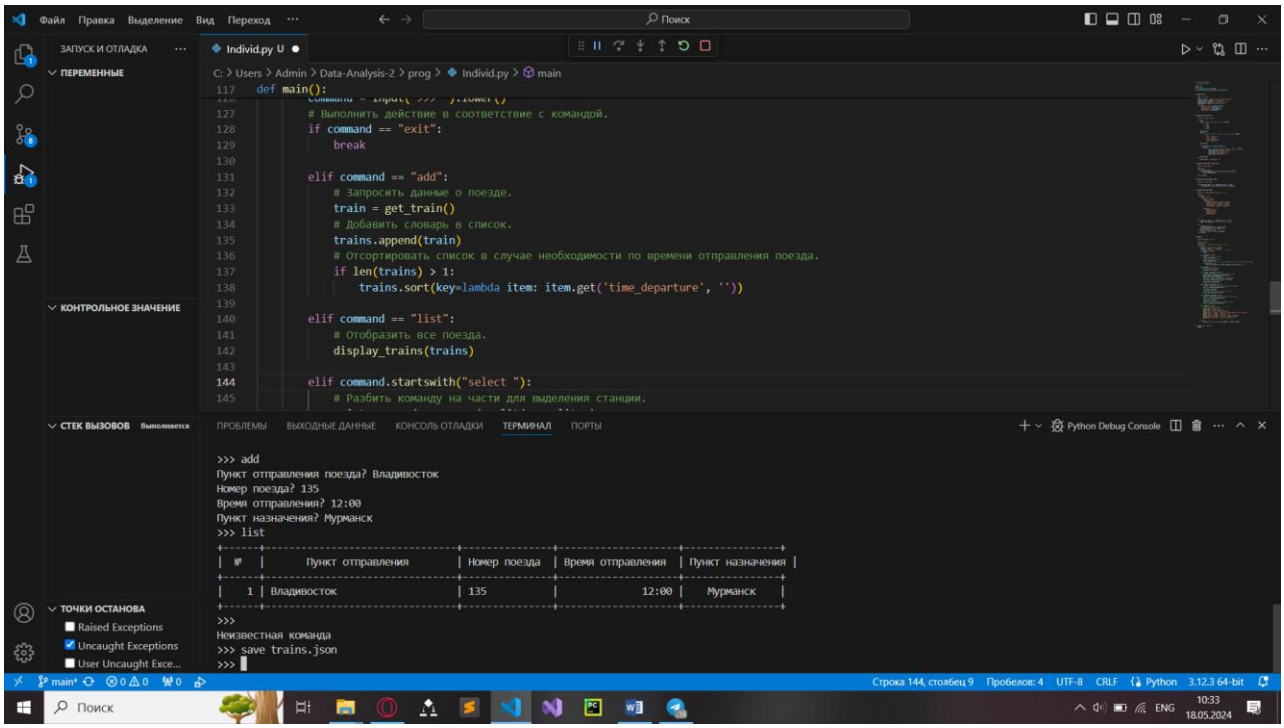


Рисунок 2. Результат работы программы из индивидуального задания 1

Задание. 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.

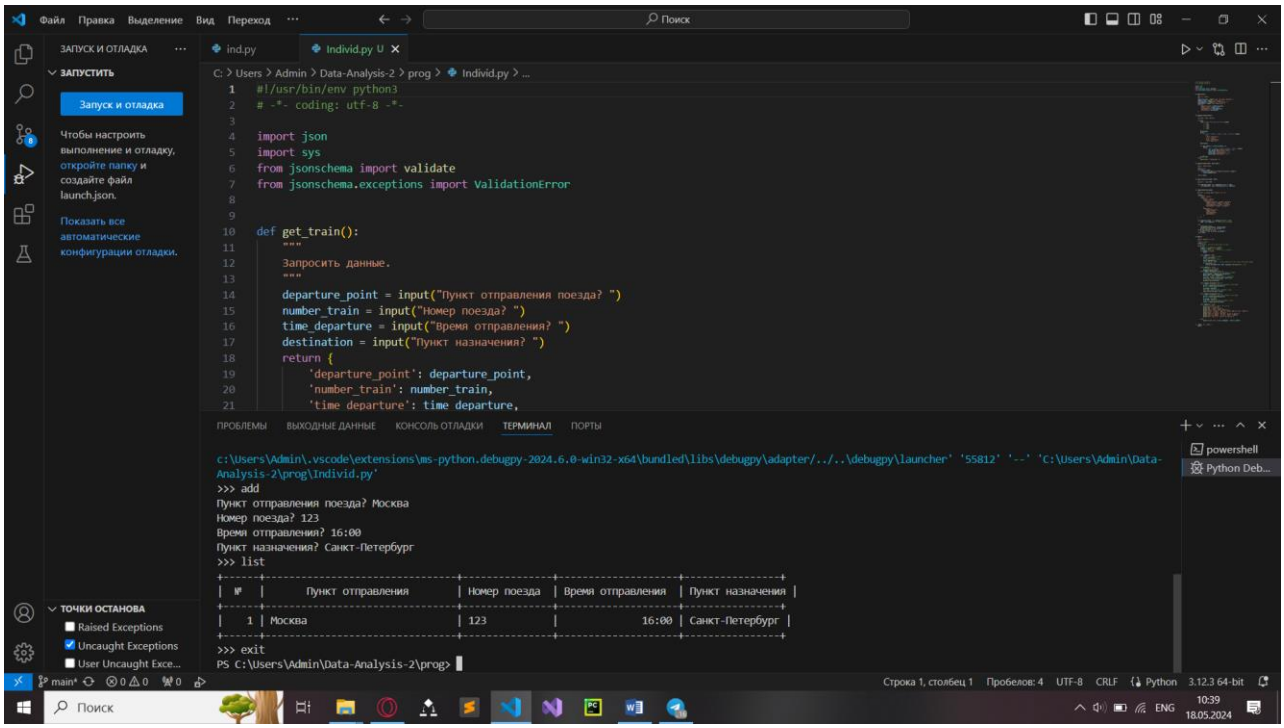


Рисунок 3. Валидация файла

## Контрольные 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.