

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ-  
СКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Основы кроссплатформенного программирования**

**Отчет по лабораторной работе №2.16**

**Тема: «Работа с данными формата JSON в языке Python»**

Выполнил студент группы

ИВТ-б-о-21-1

Артемьев Андрей В. « \_\_\_\_\_ » \_\_20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « \_\_\_\_\_ » \_\_20\_\_ г.

Проверил доцент

Кафедры инфокоммуникаций, старший  
преподаватель

Воронкин Р.А.

\_\_\_\_\_  
(подпись)

Ставрополь 2022

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

## Ход работы:

**1. Создал репозиторий в GitHub,** дополнил правила в .gitignore для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на компьютер и организовал в соответствии с моделью ветвления git-flow.

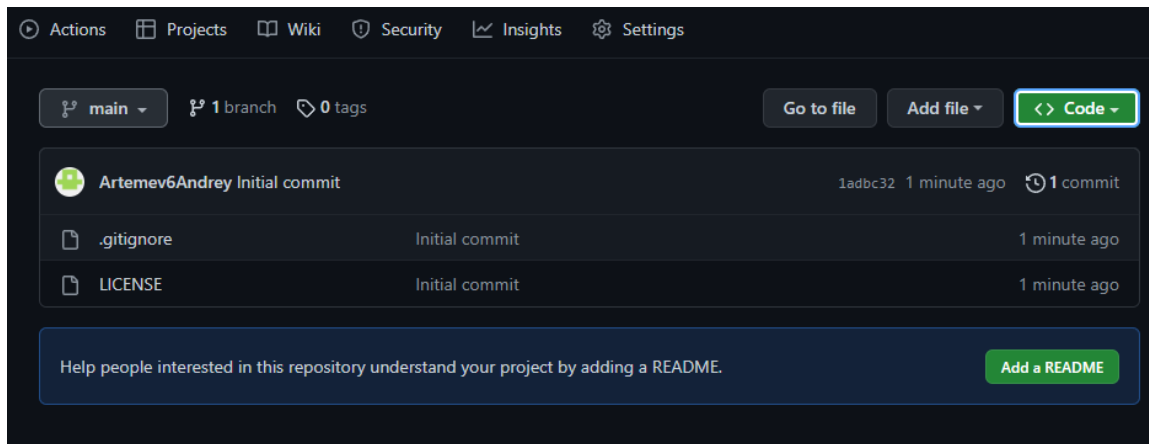


Рисунок 1.1 – Созданный репозиторий

```
.gitignore – Блокнот
Файл  Правка  Формат  Вид  Справка
.idea/
# Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
# Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm

### PyCharm ###
# Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Ar
# Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839

# User-specific stuff
.idea/**/workspace.xml
.idea/**/tasks.xml
.idea/**/usage.statistics.xml
.idea/**/dictionaries
.idea/**/shelf

# AWS User-specific
.idea/**/aws.xml

# Generated files
.idea/**/contentModel.xml
```

Рисунок 1.2 – Дополнил правила в .gitignore

```

aa715@ARTEMEV MINGW64 ~/OneDrive/Рабочий стол/11 (main)
$ git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/aa715/OneDrive/Рабочий стол/11/.git/h
]

```

Рисунок 1.3 – Организация репозитория в соответствии с моделью ветвления git-flow

2. Создал проект Pycharm в папке репозитория, проработал примеры ЛР.

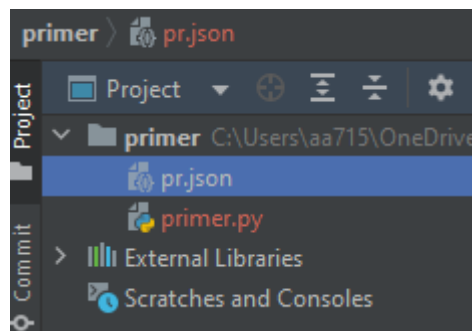


Рисунок 2 – Созданный проект

```

"C:\Users\aa715\OneDrive\Рабочий стол
>>> add
Фамилия и инициалы? ARTEMEV AV
Должность? 5
Год поступления? 21212
>>> add
Фамилия и инициалы? KRASAVA WW
Должность? 4
Год поступления? 21333
>>> save pr.json
>>>

```

Рисунок 3 – Результат работы программы

primer.py	18.12.2022 21:47	Python File
pr.json	18.12.2022 22:54	Файл "JSON"

Рисунок 4 – Созданный файл

```
>>> load pr.json
>>> select 8
```

No	Ф.И.О.	Должность	Год
1	Артемов АВ	5	2014
2	КРАСАВА ВВ	4	2012

Рисунок 5 – Загрузка данных примера

**Индивидуальное задание №1.** 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.

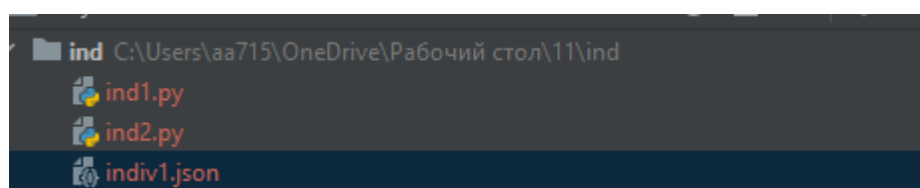


Рисунок 6 – Созданный файл

```
>>> save indiv1.json
>>> load indiv1.json
>>> select
    * artemev грyппa № ivt23
>>> list
```


№	Ф.И.О.	Группа	Успеваемость
1	artemev	ivt23	5.5.5.5.5

```
>>> |
```

Рисунок 7 – Загрузка данных из файла

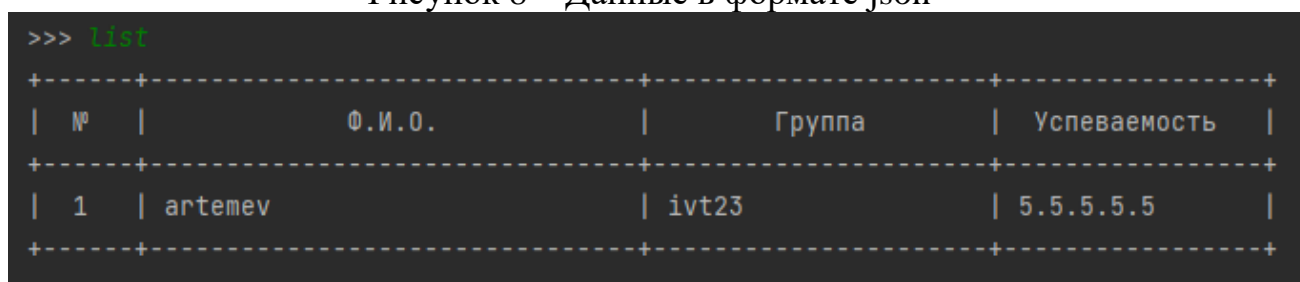
**Задание повышенной сложности.** Очевидно, что программа в примере 1 и в индивидуальном задании никак не проверяет правильность загружаемых данных формата JSON. В следствие чего, необходимо после загрузки из файла

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.



```
4      "group": "ivt23",
5      "mark": [
6          5,
7          5,
8          5,
9          5,
10         5
11     ]
12 },
13 {
14     "name": "pak",
15     "group": "ivt21",
16     "mark": [
17         5,
18         5,
19         5,
20         5,
21         5
22     ]
23 }
24 ]
25
```

Рисунок 8 – Данные в формате json



```
>>> list
```

№	Ф.И.О.	Группа	Успеваемость
1	artemev	ivt23	5.5.5.5.5

Рисунок 9 – Проверка валидации (успешная)

**Вывод:** в результате выполнения лабораторной работы были приобретение теоретические сведения и практические навыки по работе с данными формата JSON с помощью язык программирования Python версии 3.x.

### **Ответы на контрольные вопросы:**

#### **1. Для чего используется JSON?**

JSON (англ. JavaScript Object Notation, обычно произносится как JAY-сэп) – текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми.

#### **2. Какие типы значений используются в JSON?**

Набор пар ключ: значение. Упорядоченный набор значений.

#### **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 обмена данными JSON5 (JSON5) - это надмножество JSON, целью которого является смягчение некоторых ограничений JSON путем расширения его синтаксиса для включения некоторых продуктов из ECMAScript 5.1. Эта библиотека JavaScript является официальной эталонной реализацией библиотек синтаксического анализа и сериализации JSON5.

Краткое описание возможностей. Следующие функции ECMAScript 5.1, которые не поддерживаются в JSON, были расширены до JSON5. Объекты.

Ключи объекта могут быть идентификатором ECMAScript 5.1.

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

Реализация Python формата данных JSON5.

JSON5 расширяет формат обмена данными JSON, чтобы сделать его более удобным для использования в качестве языка конфигурации:

Комментарии в стиле JavaScript (как однострочные, так и многострочные) разрешены.

Ключи объектов могут быть без кавычек, если они являются допустимыми идентификаторами ECMAScript.

Объекты и массивы могут заканчиваться запятыми.

Строки могут заключаться в одинарные кавычки, допускаются многострочные строковые литералы.

Есть еще несколько более мелких расширений JSON; см. полную информацию на странице выше.

Этот проект реализует реализацию чтения и записи для Python; где возможно, он отражает стандартный пакет Python JSON API для простоты использования.

Есть одно заметное отличие от JSON api: методы `load ()` и `load ()` поддерживают опциональную проверку (и отклонение) повторяющихся ключей объекта; `pass allow_duplicate_keys = False` для этого (по умолчанию разрешены дубликаты).

Это ранний выпуск. Это было достаточно хорошо протестировано, но это *МЕДЛЕННО*. Он может быть в 1000-6000 раз медленнее, чем модуль JSON, оптимизированный для C, и в 200 раз (или более) медленнее, чем модуль JSON на чистом Python.

## **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.

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

Dumps записывает в строку, а dump в файл.

## **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 (s на конце от string/строка)

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

`import codecs`

`json.load(codecs.open('sample.json', 'r', 'utf-8-sig'))`

## **10. Самостоятельно ознакомьтесь со спецификацией JSON Schema?**

### **Что такое схема данных?**

Схема JSON - это словарь, который позволяет аннотировать и проверять документы JSON.

Преимущества:

- Описывает ваш существующий формат (ы) данных.
- Предоставляет понятную документацию, читаемую человеком и машиной.
- Проверяет данные, которые полезны для:
- Автоматизированное тестирование.
- Обеспечение качества предоставленных клиентом данных.