МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №2.16

Дисциплина: «Программирование на Python»

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

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

Кочкаров Умар Ахматович

Ставрополь 2022

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

Ход работы:

1. Создать общедоступный репозиторий с лицензией MIT и языком Python.

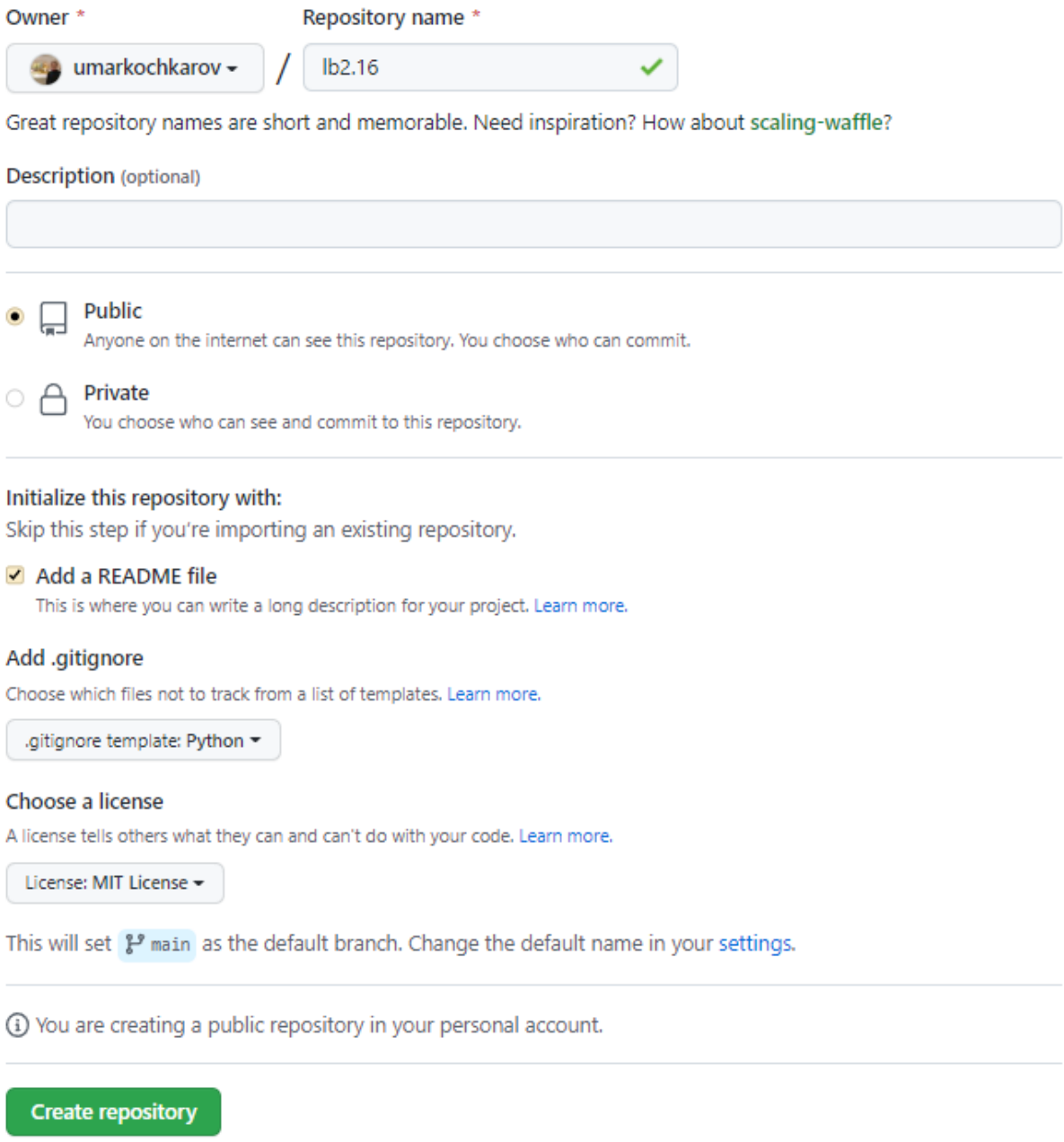

Рисунок 1. Создание репозитория

2. Клонировать репозиторий на ПК:

```
erken@LAPTOP-ESTC6OGF MINGW64 ~/Desktop/python/Ла62.16
$ git clone https://github.com/umarkochkarov/lb2.16.git
Cloning into 'lb2.16'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 2. Клонирование репозитория

3. Организовать репозиторий в соответствии с моделью ветвления git-flow.

```
erken@LAPTOP-ESTC6OGF MINGW64 ~/Desktop/python/Ла62.16/lb2.16 (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/erken/Desktop/python/Ла62.16/lb2.16/.git/
hooks]
```

Рисунок 3. Организация репозитория в соответствии с моделью git-flow

4. Проработка примера из лабораторной работы:

```
C:\Users\erken\Desktop\python\Ла62.16\lb2.16\env\Scripts\python.exe C:/Users/erken/Desktop/python
>>> add
Фамилия и инициалы? abu aa
Должность? dir
Год поступления? 2010
>>> add
Фамилия и инициалы? bandit bb
Должность? man
Год поступления? 2014
>>> list
+------+----------------------------------+----------------------+----------+
|  No  |            Ф.И.О.                |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | abu aa                           | dir                  |     2010 |
|    2 | bandit bb                        | man                  |     2014 |
+------+----------------------------------+----------------------+----------+
>>> save prim
>>> load prim
>>> exit
```

Рисунок 4. Пример

5. Индивидуальные задания

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.

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-sch ema.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.

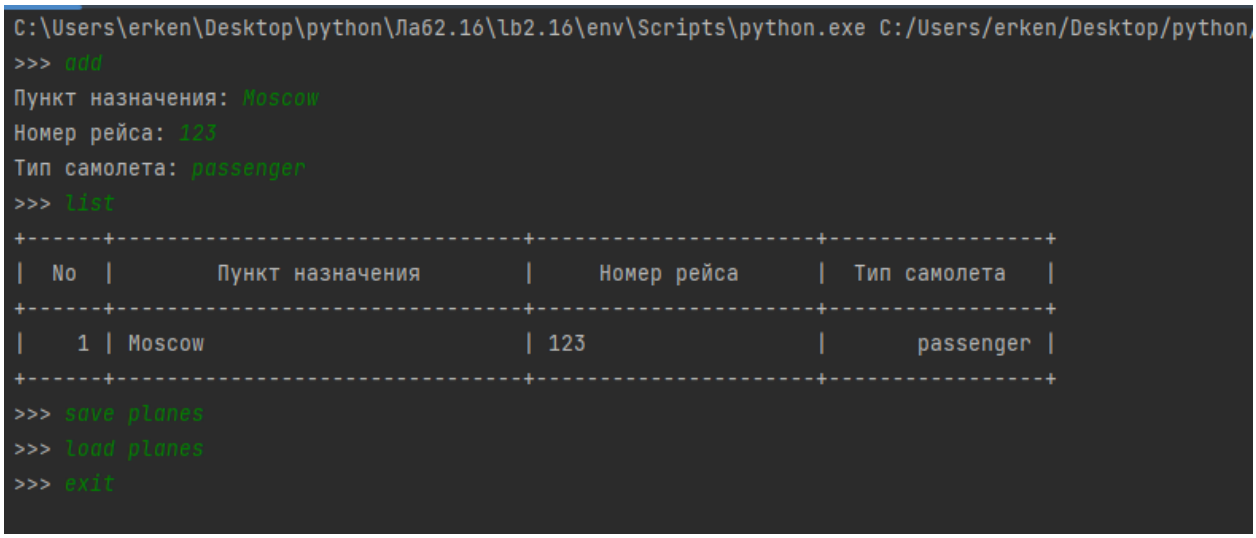

Рисунок 5. Индивидуальное задание

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. Это информативное руководство поможет вам быстрее разобраться с данными, которые вы можете использовать с JSON и основной структурой с синтаксисом этого же формата.

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 имволов 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.Использование вложенности в нашем JSON формате позволяет нам работать с наиболее сложными и иерархичными данными.

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

JSON5 — это расширение популярного формата файлов JSON, которое упрощает написание и поддержку вручную (например, для файлов конфигурации). Он не предназначен для межмашинного взаимодействия.

Формально формат обмена данными JSON5 является расширенным набором JSON (поэтому допустимые файлы JSON всегда будут действительными файлами JSON5), который расширяет свой синтаксис, чтобы включить некоторые продукты из ECMAScript 5.1 (ES5).Это также строгое подмножество ES5, поэтому действительные файлы JSON5 всегда будут действительными ES5.

5. Какие средства языка программирования Python могут быть использованы для работы с данными в формате 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 редставление. 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 (s на конце от string/строка).

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.

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

json.dump(staff, fout, ensure_ascii=False, indent=4).

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

JSON Schema — один из языков описания структуры JSON-документа. Использует синтаксис JSON. Базируется на концепциях XML Schema, RelaxNG, Kwalify. JSON Schema — самоописательный язык: при его использовании для обработки данных и описания их допустимости могут использоваться одни и те же инструменты сериализации / десериализации.

Схема данных - - модель объекта, представленная в виде информации, описывающей существенные для данного рассмотрения параметры и переменные величины объекта, связи между ними, входы и выходы объекта и позволяющая путём подачи на модель информации об изменениях входных величин моделировать возможные состояния объекта.

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