МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ» ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №4 по дисциплине «Основы программной инженерии»

Выполнил студент

2 курса, группы ПИЖ-б-о-20-1

Тотубалина С.С.

Проверил:

Доцент кафедры инфокоммуникаций,

Воронкин Р.А.

Ставрополь, 2022 г

Ход работы

1. Проработка примеров лабораторной работы.

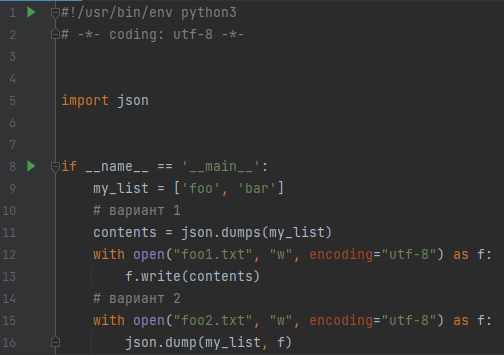


Рисунок 1.1 – код программы ex1.py

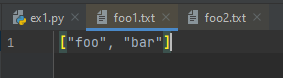
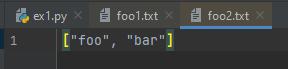
 

Рисунок 1.2 – результат работы программы ex1.py

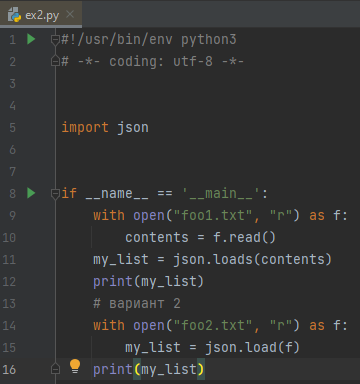


Рисунок 1.3 – код программы ex2.py

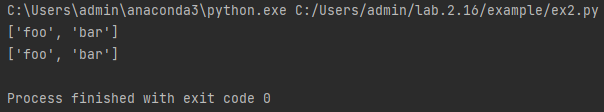


Рисунок 1.4 – результат работы программы ex2.py

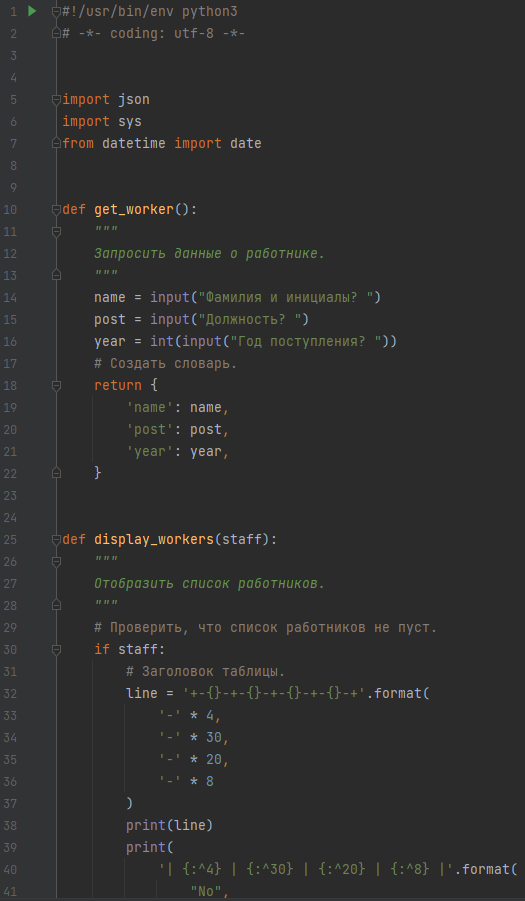


Рисунок 1.5 – код программы ex3.py

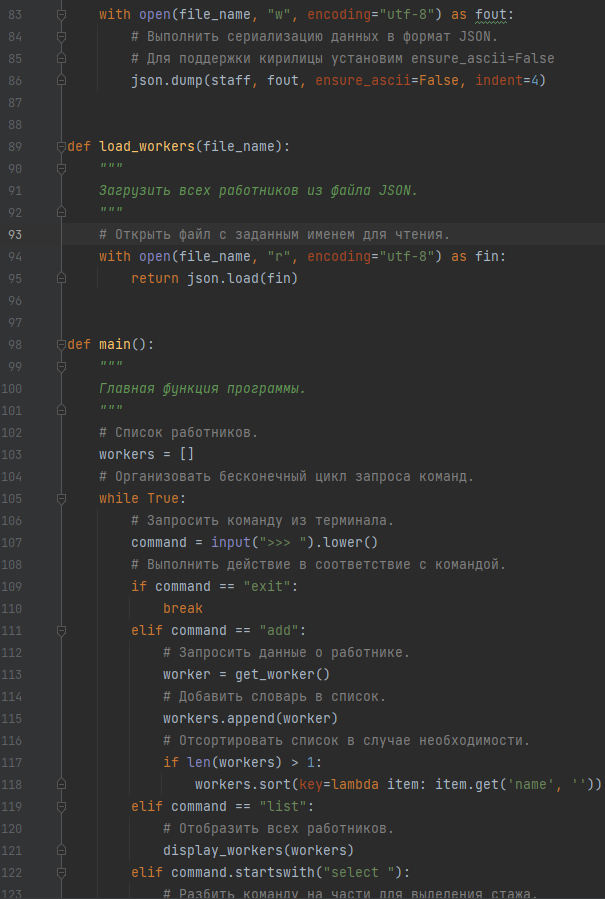


Рисунок 1.6 – код программы ex3.py

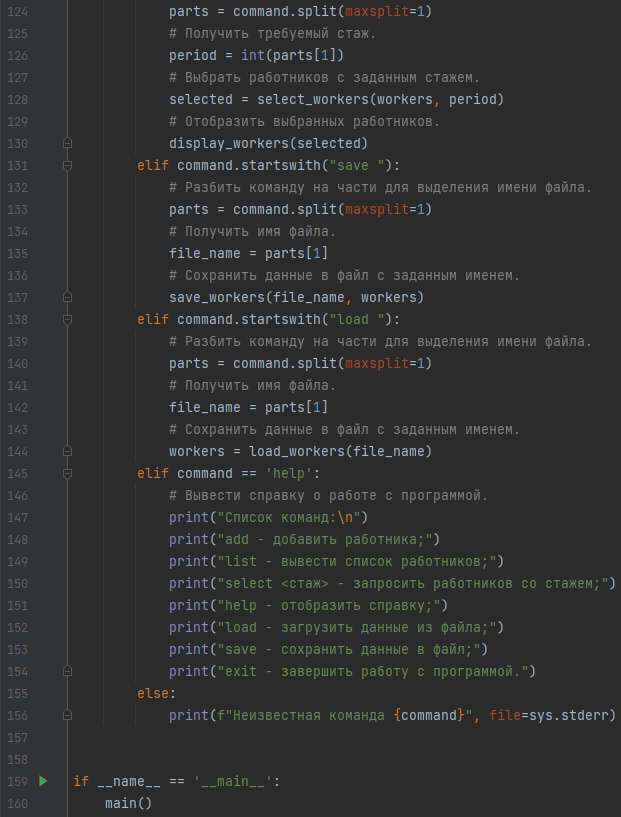


Рисунок 1.7 – код программы ex3.py

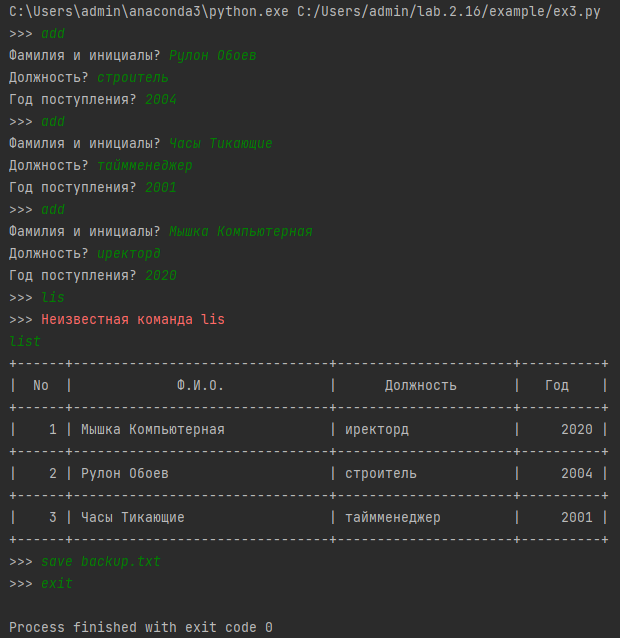


Рисунок 1.8 – результат работы программы ex3.py

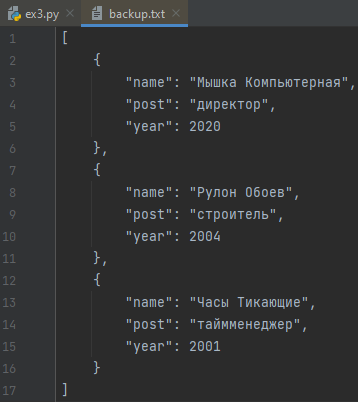


Рисунок 1.9 – результат работы программы ex3.py

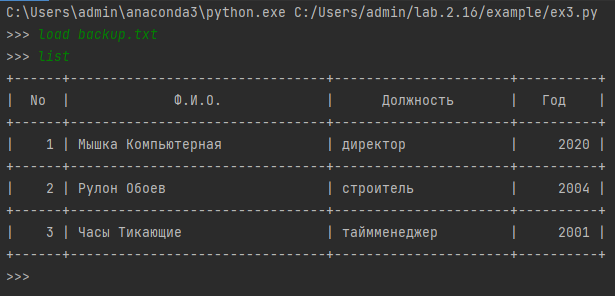


Рисунок 1.10 – результат работы программы ex3.py

2. Индивидуальное задание.



Рисунок 2.1 – код программы ind\_task.py



Рисунок 2.2 – код программы ind\_task.py



Рисунок 2.3 – код программы ind\_task.py

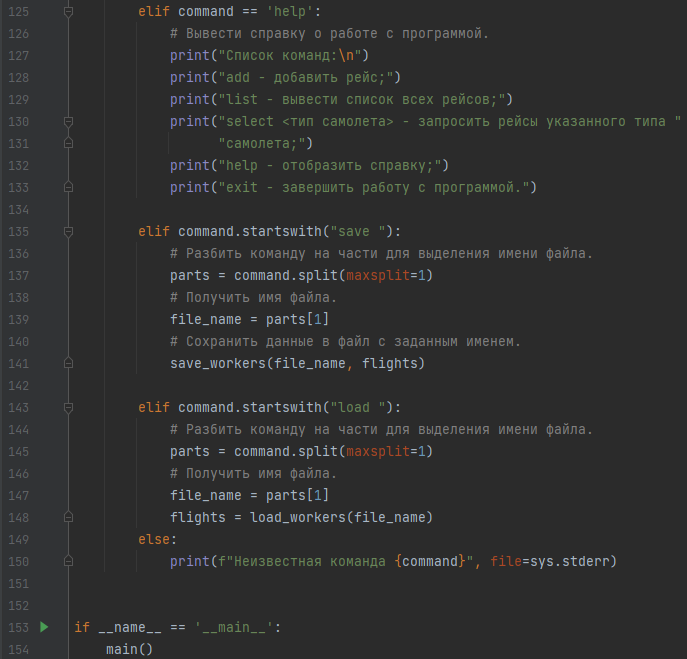


Рисунок 2.4 – код программы ind\_task.py

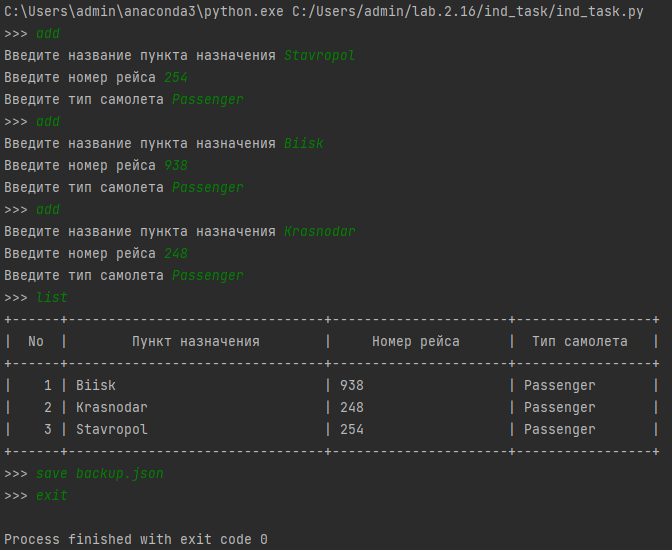


Рисунок 2.5 – результат работы программы ind\_task.py

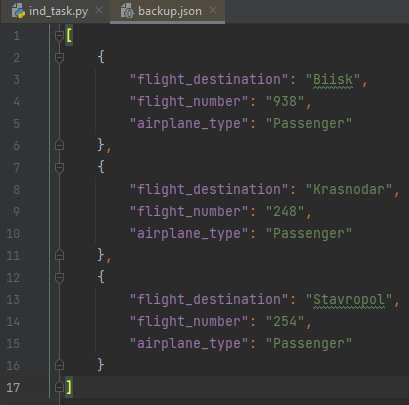


Рисунок 2.6 – файл backup.json

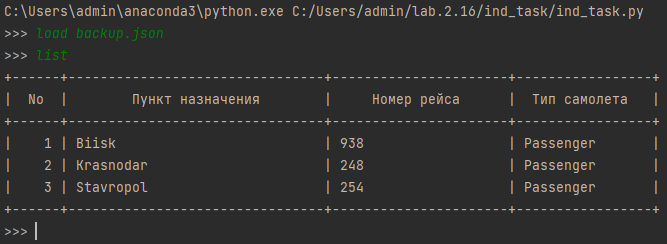


Рисунок 2.7 – результат работы программы ind\_task.py

Ответы на вопросы:

1. Для чего используется 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.

Объект JSON это формат данных — ключ-значение, который обычно рендерится в фигурных скобках. Когда вы работаете с JSON, то вы скорее всего видите JSON объекты в .json файле, но они также могут быть и как JSON объект или строка уже в контексте самой программы.

2. Какие типы значений используются в JSON?

Если быть точным, то им нужно быть одним из шести типов данных: строкой, числом, объектом, массивом, булевым значением или null.

Как было показано ранее JSON-текст представляет собой (в закодированном виде) одну из двух структур:

Набор пар ключ: значение. В различных языках это реализовано как запись, структура, словарь, хеш-таблица, список с ключом или ассоциативный массив. Ключом может быть только строка (регистрозависимость не регулируется стандартом, это остаётся на усмотрение программного обеспечения. Как правило, регистр учитывается программами — имена с буквами в разных регистрах считаются разными, значением — любая форма. Повторяющиеся имена ключей допустимы, но не рекомендуются стандартом; обработка таких ситуаций происходит на усмотрение программного обеспечения, возможные варианты — учитывать только первый такой ключ, учитывать только последний такой ключ, генерировать ошибку.

Упорядоченный набор значений. Во многих языках это реализовано как массив, вектор, список или последовательность.

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.

3. Как организована работа со сложными данными в JSON?

Вложенные объекты

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. Фигурные скобки везде используются для формирования, вложенного JSON объекта с ассоциированными именами пользователей и данными локаций для каждого из них. Как и с любым другим значением, используя объекты, двоеточие используется для разделения элементов.

{

"sammy" : {

username" : "SammyShark",

"location" : "Indian Ocean",

"online" : true,

"followers" : 987

},

"jesse" : {

"username" : "JesseOctopus",

"location" : "Pacific Ocean",

"online" : false,

"followers" : 432

},

"drew" : {

"username" : "DrewSquid",

"location" : "Atlantic Ocean",

"online" : false,

"followers" : 321

},

"jamie" : {

"username" : "JamieMantisShrimp",

"location" : "Pacific Ocean",

"online" : true,

"followers" : 654

}

}

Вложенные массивы

Данные также могут быть вложены в формате JSON, используя JavaScript массивы, которые передаются как значения. JavaScript использует квадратные скобки [ ] для формирования массива. Массивы по своей сути — это упорядоченные коллекции и могут включать в себя значения совершенно разных типов данных. Мы можем использовать массив при работе с большим количеством данных, которые могут быть легко сгруппированны вместе, как например, если есть несколько разных сайтов и профайлов в социальных сетях ассоциированных с одним пользователем.

{

"first\_name" : "Sammy",

"last\_name" : "Shark",

"location" : "Ocean",

"websites" : [

{

"description" : "work",

"URL" : "https://www.digitalocean.com/"

},

{

"desciption" : "tutorials",

"URL" : "https://www.digitalocean.com/community/tutorials"

}

],

"social\_media" : [

{

"description" : "twitter",

"link" : "https://twitter.com/digitalocean"

},

{

"description" : "facebook",

"link" : "https://www.facebook.com/DigitalOceanCloudHosting"

},

{

"description" : "github",

"link" : "https://github.com/digitalocean"

}

]

}

Ключи "websites" и "social\_media" используют массив для вложения информации о сайтах пользователя и профайлов в социальных сетях. Мы знаем, что это массивы — из-за квадратных скобок.

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 — предложенное расширение формата json в соответствии с синтаксисом ECMAScript 5, вызванное тем, что json используется не только для общения между программами, но и создаётся/редактируется вручную. Файл JSON5 всегда является корректным кодом ECMAScript 5. JSON5 обратно совместим с JSON. Для некоторых языков программирования уже существуют парсеры json5.

Некоторые нововведения:

Поддерживаются как однострочные //, так и многострочные /\* \*/ комментарии.

Записи и списки могут иметь запятую после последнего элемента (удобно при копировании элементов).

Ключи записей могут быть без кавычек, если они являются валидными идентификаторами ECMAScript 5.

Строки могут заключаться как в одинарные, так и в двойные кавычки.

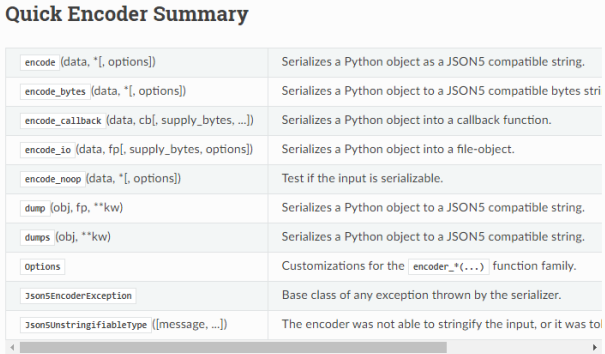
Числа могут быть в шестнадцатеричном виде, начинаться или заканчиваться десятичной точкой, включать Infinity, -Infinity, NaN и -NaN, начинаться со знака +.

Проще говоря, он убирает некоторые ограничения JSON, расширяя его синтаксис.

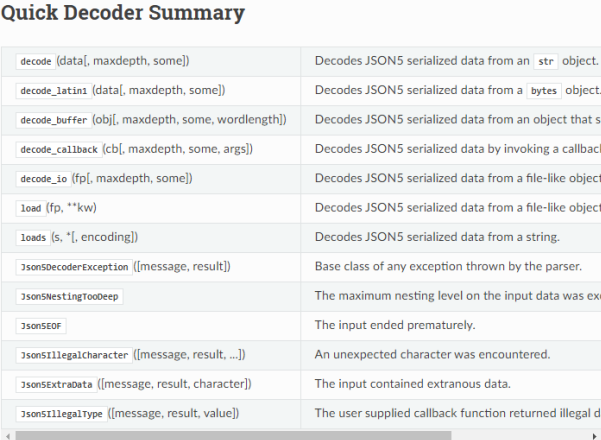
5. Какие средства языка программирования Python могут быть использованы для работы с данными в формате JSON5?

Существует пакет PyJSON5, который содержит множество функций для расширения функционала JSON.

Ниже представлены функции для сериализации данных



Функции для кодирования/декодирования данных:



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

Сериализация данных в формат 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.

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

json.dumps() конвертирует python объект в json и записывает его в строку вместо записи в файл.

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

Десериализация 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.

Если не удастся десериализовать JSON, будет возбуждено исключение ValueError .

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

Использование кодировки UTF-8 или ensure\_ascii=False

10. Самостоятельно ознакомьтесь со спецификацией JSON Schema? Что такое схема данных?

Приведите схему данных для примера 1.

Схема данных представляет собой код, который используется для валидации данных в формате JSON. Схема данных:

