**Тема занятия №14: Работа с комплексными файлами - excel, json, word. Библиотеки openpyxl, json, docx**

**1. Работа с библиотекой json – открытие, чтение и запись**

Python поддерживает JSON

Python содержит встроенный модуль под названием json для кодирования и декодирования данных JSON.



Небольшой словарь

Как правило, процесс кодирования JSON называется сериализация. Этот термин обозначает трансформацию данных в серию байтов (следовательно, серийных) для хранения или передачи по сети. Также вы, возможно, уже слышали о термине «маршалинг», но это уже совсем другая область.

Естественно, десериализация — является противоположным процессом декодирования данных, которые хранятся или направлены в стандарт JSON.

Сериализация JSON

Что происходит после того, как компьютер обрабатывает большие объемы информации? Ему нужно принять дамп данных. Соответственно, модуль json предоставляет метод dump() для записи данных в файлы. Также есть метод dumps() для записей в строку Python.

Простые объекты Python переводятся в JSON согласно с весьма интуитивной конверсией.

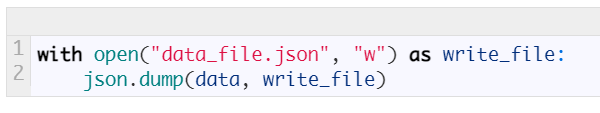


Пример сериализации JSON Python



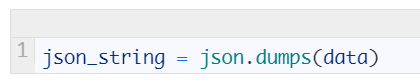
Сохранить эту информацию на диск — критично, так что ваша задача — записать на файл.

Используя контекстный менеджер Python, вы можете создать файл под названием data\_file.json и открыть его в режиме write (файлы JSON имеют расширение .json).



Обратите внимание на то, что dump() принимает два позиционных аргумента: (1) объект данных, который сериализуется и (2), файловый объект, в который будут вписаны байты.

Или, если вы склонны продолжать использовать эти сериалзированные данные JSON в вашей программе, вы можете работать как со строкой.



Обратите внимание, что файловый объект является пустым, так как вы на самом деле не выполняете запись на диск. Кроме того, dumps() аналогичен dump().