***Можливості пакету json***

**JSON** — це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дозволяє описувати об'єкти та інші структури даних. Цей формат головним чином використовується для передачі структурованої інформації через мережу.

Щоб використати пакет JSON потрібно зробити імпорт.

C:\Users\NIkita\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_8.png

JSON має багато функції для використвання.

***json.dump***(obj, fp, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True,  cls=None, indent=None, separators=None, encoding="utf-8", default=None, sort\_keys=False, \*\*kw) – робить серіалізацію як відформатований JSON потік.

Якщо **skipkeys** = True, тоді ключі словника, які не є базового типу(*str, Unicode, int, long, float, bool, None*) будуть видавати помилку *TypeError.*

Якщо **ensure\_ascii=True**, то всі ASCII символи будуть у виводі.

Якщо **check\_circular=False**, то перевірка циклічних посилань буде пропущена, а такі посилання викликатимуть *OverflowError*.

Якщо **allow\_nan=False**, при спробі серіалізувати значення з комою, яка виходить за допустимі межі, тоді буде збуджуватись помилка *ValueError* у строгій відповідності зі специфікацією JSON.

Якщо **indent** - невід’ємне число або рядок, то масиви і об’єкти в JSON будуть виводитися з таким рівнем відступу. Якщо рівень відступу 0, від’ємний або «», то будуть просто використовуватись нові рядки. Значення по замовчуванню (None) вибирає найбільш компактне відображення. Якщо indent – це рядок ( такий як ‘\t’), то цей рядок використовуватись у якості відступу.

Якщо **sort\_keys=True**, то словник відсортується за ключем.

***json.dumps*** (obj, fp, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, cls=None, indent=None, separators=None, default=None, sort\_keys=False, \*\*kw) – робить серіалізацію obj в *str* JSON-формату.

Якщо **ensure\_ascii = False**,то результат має включати ASCII символи і повертає значення Unicode приклад

Всі інші аргументи мають такі ж самі значення, що і в **json.dump()**

***json.load***(fp[, encoding[, cls[, object\_hook[, parse\_float[, parse\_int[, parse\_constant[, object\_pairs\_hook[, \*\*kw]]]]]]]]) – робить десеріалізацію у *fp*

**object\_hook** - це додаткова функція, яка виконується до результату декодування об’єкта(*dict*). *object.hook* буде використовуватися замість *dict*

**object\_pairs\_hook**  - це дотаткова функція, яка виконується до результату декодування з упорядкованим список пар. *object\_pairs\_hook* буде використовуватися замість *dict*.

***json.loads***( s, encoding=None, object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, object\_pairs\_hook=None, \*\*kw) – робить дисеріалізацію s (екземпляр *str*, який містить документ JSON) в об’єкт Python використовувавши перекладну таблицю .

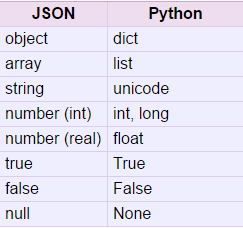
C:\Users\NIkita\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_10.png

Всі інші аргументи мають такі ж самі значення, що і в **json.load()**

***Кодувальники та декодувальники.***

***Class json.JSONDecoder***( object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, strict=True, object\_pairs\_hook=None) – простий кодувальник JSON.

Виконує наступні перетворення при декодуванні:

******

**object\_hook** - якщо визначний , буде викликаний для результату декодування кожного JSON об’єкту декодованого значення та буде використовуватися замість даного *dict.*

**object\_pairs\_hook**  - якщо визначний, буде викликаний для результату декодування кожного JSON об’єкта з упорядкованим список пар. *object\_pairs\_hook* буде використовуватися замість *dict*.

**parse\_float** - якщо визначений, буде викликаний для кожного значення JSON з плаваючою крапкою для його декодування. За замовчуванням це еквівалентно *float(num\_str).*

***parse\_int***, якщо визначений, буде викликаний для рядка з числовим значенням JSON, щоб його декодувати.

***parse\_constant*** – якщо визначений, буде викликаний одним з наступних рядків:

 '-Infinity', 'Infinity', 'NaN'.

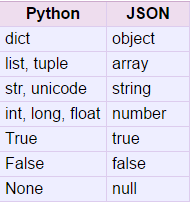
Якщо **strict=False** ( по замовчуванню **True**), тоді всередині рядка можуть використовуватись керуючі символи. В даному контексті керуючі символи – це символи з кодами в інтервалі 0-31, включаючи ‘\t’ (tab), ‘\n’, ‘\r’, ‘\0’.

***decode(s)*** – повертає представлення Python для s ( *str* або *Unicode,* що міститься у JSON документа)

***raw\_decode(s)*** – декодує JSON документ з s ( *str* або *Unicode*  починається з JSON документа) і повертає кортеж з 2 елементів:

1. представлення Python
2. індекс рядка (s), на якому документ закінчується.

**class*json.JSONEncoder***([skipkeys[, ensure\_ascii[, check\_circular[, allow\_nan[, sort\_keys[, indent[, separators[, encoding[, default]]]]]]]]])



Якщо **skipkeys** **= False,** збуджується помилка *TypeError*, при спробі закодувати ключ, який не являється *str, int, long, float* or *None*.

Якщо **skipkeys = True**, деякі об’єкти будять ігноруватися.

Якщо **ensure\_ascii=True**, то всі не ASCII символи у виводі екрануються послідовностями *\uXXXX* і результатом є рядок, який містить лише ASCII символи.

Якщо **check\_circular=True**, тоді списки, словники і вручну закодовані об’єкти перевіряються на циклічні посилання при кодуванні для запобігання нескінченної рекурсії (що може призвести до *OverflowError*).

Якщо **sort\_keys=True**, то словник, що виводиться, буде відсортований по ключах.

Якщо **indent** невід’ємне число, тоді масиви та об’єкти JSON будуть виведені з цим рівнем відступу.

Якщо задано, то **default** – це функція, яка викликається для об’єктів, які були серіалізовані. Вона повинна повертати JSON кодовану версію для об’єкту або збуджувати *TypeError*.

Якщо **encoding** не *None*, то всі рядки, які вводяться будуть перетворені в юнікод, використовуючи це кодування.

**encode(o)** – повертає рядкове представлення JSON для структури даних Python o.

**Iterencode(o)** – кодує переданий об’єкт о і видає кожне рядкове представлення, як тільки воно стає доступним.