***2 етап виконання курсової роботи***

***Вивчення засобів стандартної бібліотеки Python JSON для роботи з даними різних типів***

**JSON** — це текстовий формат обміну даними між комп'ютерами. JSON базується на мові програмування JavaScript. Формат дозволяє описувати об'єкти та інші структури даних.

*Основне використання.*

json.dump(*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*)

- Якщо skipkeys є True (за замовчуванням False), тоді ключі словника, які не є базового типу (str, int, float, bool, None) будуть пропущені замість виведення помилки TypeError.

- Модуль json завжди справляє об’єкти str, а не bytes. Тому, fp.white() повинен підтримувати str ввід.

- Якщо ensure\_ascii є True (за замовчуванням), то вивід гарантовано пропустить всі вхідні елементи, які не є ASCII. Якщо ж ensure\_ascii є False, то ці елементи будуть виведені в первинному вигляді.

- Якщо check-circular є False (за замовчуванням True), то циклічна перевірка типів контейнерів буде пропущена і циклічне посилання приведе до OverflowError (чи гірше).

- Якщо allow\_nan є False (за замовчуванням True), тоді буде ValueError, щоб серіалізувати значення float (nan, inf, -inf) поза межею (out of range) в суворій відповідності специфікації JSON. Якщо ж allow\_nan є True, то еквіваленти JavaScript будуть використані (NaN, Infinity, -Infinity).

- Якщо indent є невід’ємним цілим або з плаваючою крапкою числом (int or float), то елементи масиву JSON і члени об’єкту будуть досить віддрукованими з цим рівнем відступу. Рівень відступу 0, від’ємний, чи “” буде вставляти нові рядки. None (за замовчуванням) вибирає найбільш компактну подачу. Використання додатного цілого числа відступів відступає стільки пробілів на кожен рівень. Якщо indent є рядком (наприклад, “\t”), то цей рядок використовується для відступу на кожному рівні.

- Якщо вказано, то separators повинні бути (item\_separator, key\_separator) кортежем (tuple). За замовчуваннням – (‘, ‘, ‘: ‘), якщо indent є None та (‘,‘, ‘:‘) інакше. Щоб отримати найбільш компактне JSON представлення, слід вказати (‘,‘, ‘:‘) для усунення пробілів.

- Якщо вказано, default повинен бути функцією, що може бути викликана для об’єктів, які в іншому випадку не можуть бути серіалізовані. Це повинно повертати закодовану версію JSON об’єкту або викликати TypeError. Якщо не вказано, TypeError викликається сама.

- Якщо sort\_keys є True (за замовчуванням False), то вивід словника буде відсортований за ключем.

json.dumps(obj, skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, cls=None, indent=None, separators=None, default=None, sort\_keys=False, \*\*kw) - серіалізує obj в рядок JSON-формату. Аргументи мають те саме значення, що і для dump().

json.load(fp, cls=None, object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, object\_pairs\_hook=None, \*\*kw)

- object\_hook – це необов’язкова функція яка буде викликана до результату будь-якого розкодованого об’єкту (a dict). Значення, що повертається буде використане замість словника (dict).

- object\_pairs\_hook – це необов’язкова функція що буде викликана до результату будь-якого декодованого об’єкту з впорядкованим списком пар (ключ і значення). Значення, що повертається буде використане замість словника (dict). Якщо задано також object\_hook, то пріотритет буде надано object\_pairs\_hook.

- parse\_float – якщо визначений, то буде викликатися з рядком кожного JSON значення з плаваючою крапкою. За замовчуванням, це еквівалентно до float(num\_str). Це може бути використано для використання іншого типу даних.

-parse\_int – якщо визначений, то буде викликатися з рядком кожного JSON значення цілого числа. За замовчуванням, це еквівалентно до int(num\_str). Це може бути використано для використання іншого типу даних.

- parse\_constant – якщо визначений, то буде викликаний з одним з наступних рядків: ‘-Infinity’, ‘Infinity’, ‘NaN’. Це також може бути використано для підняття виключення якщо в форматі JSON зустрічаються недійсні номери.

Якщо не вдається дисеріалізувати JSON, то буде збуджена виняток ValueError.

json.loads( s, encoding=None, object\_hook=None, parse\_float=None, parse\_int=None, parse\_constant=None, object\_pairs\_hook=None, \*\*kw) – дисеріалізує s (екземпляр str, який містить документ JSON) в об’єкт Python. Решта аргументів мають те саме значення, що і в json.load(), крім encoding, який ігнорується.

Якщо дані, які були дисеріалізовані, має недійсне значення, то буде збуджений виняток JSONDecodeError.

*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 декодувальник JSON. Виконує наступні дії при декодуванні (JSON в Python): object -> dict, array -> list, string -> str, number(int) -> int, number(real) -> float, true -> True, false -> False, null -> None. Також розуміє -Infinity, Infinity, NaN як відповідні значення з плаваючою точкою, яка знаходиться за межами специфікації JSON.

-object\_hook – якщо визначений, буде викликатися для результату декодування кожного JSON об’єкту і повернене ним значення буде використане замість декодованого значення.

-object\_pairs\_hook – якщо визначений, буде викликатися для декодованого об’єкту JSON із впорядкованим списком пар. Повернене значення цього методу буде використовуватись замість декодованого словника (dict).

-parse\_float – якщо визначений, буде викликатися для кожного значення JSON з плаваючою крапкою для його декодування.

-parse\_int – якщо визначений, буде викликатися для рядка кожного значення JSON цілого числа, щоб його декодувати.

-Якщо strict є false (True за замовчуванням), тоді керуючи символи можна використовувати всередині рядка. Керуючі символи означають символи з кодами на проміжку від 0 до 31, включно з ‘\t’ (tab), ‘\n’, ‘\r’ та ‘\0’.

-Якщо серіалізована інформація не є потрібним (дійсним) JSON документом, то JSONDecodeError буде викликана.

decode(s) – повертає представлення s у Python (str значення, що містить JSON документ).

Буде збуджено JSONDecodeError, якщо даний JSON документ недійсний.

raw\_decode(s) – декодує JSON документ з s (str починається з JSON документу) і повертає двоелементний кортеж (2-tuple) – представлення Python та індекс s, де закінчується документ.

Це може бути використане для декодування документу JSON з рядку, що може мати зайві дані в кінці.

class json.JSONEncoder(skipkeys=False, ensure\_ascii=True, check\_circular=True, allow\_nan=True, sort\_keys=False, indent=None, separators=None, default=None) – гнучкий JSON кодер структур даних Python. (Python -> JSON): dict -> object, list (tuple) -> array, str-> string, int (long, float) -> number, True -> true, False -> false, None -> null.

-Якщо skipkeys є false (за замовчуванням), тоді виде TypeError щоб досягнути кодування ключів, що не є str, int, float, None. Якщо skipkeys є true, то такі значення просто пропускаються.

-Якщо ensure\_ascii є true (за замовчуванням), то вихідні дані гарантовано не міститимуть пропущені вхідні елементи не ASCII. Якщо ensure\_ascii є false, то дані вхідні елементи будуть виведені в первинному вигляді.

-Якщо check\_circular є true (за замовчуванням), то списки, словники і кодовані елементи користувача будуть перевірені на певні відповідності впродовж кодування, для того щоб запобігти нескінченній рекурсії (яка призведе до OverflowError). В іншому випадку, ніякої перевірки не потрібно.

-Якщо allow\_nan є true (за замовчуванням), тоді Nan, Infinity та –Infinity будуть так само кодовані.

-Якщо sort\_keys є true (за замовчуванням False), тоді вивід словників буде сортуватися за ключем.

-Якщо indent – це невід’ємне ціле число або рядок, тоді елементи масиву JSON та об’єкти JSON будуть виведені за цим рівнем доступу.

-Якщо вказано, то separators повинні бути кортежем (item\_separator, key\_separator). ). За замовчуваннням – (‘, ‘, ‘: ‘), якщо indent є None та (‘,‘, ‘:‘) інакше.

-Якщо вказано, default повинен бути функцією що може бути викликана об’єктами, що в іншому випадку не можуть бути серіалізовані. Вона повинна повертати JSON кодовану версію для об’єкту або збуджувати TypeError.

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.