1. **Parse JSON (JSON 🡪 Python)**

JSON chỉ là **một chuỗi văn bản**. Nếu không chuyển thành đối tượng Python, bạn **không thể truy cập, thay đổi, tính toán hay sử dụng dữ liệu đó** trong chương trình được.

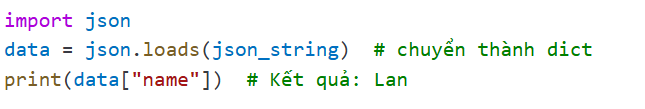
Vd:



Nếu để nguyên chuỗi, bạn **không thể truy cập name hay age**.

Nhưng sau khi chuyển:

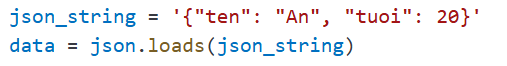
Ta dùng JSON.loads() để làm điều này:



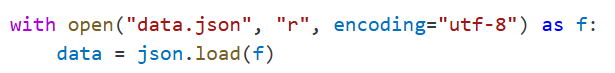
🡪Chuyển chuỗi JSON thành đối tượng Python là bước bắt buộc để bạn có thể *hiểu, sử dụng và thao tác với dữ liệu* trong chương trình Python một cách hiệu quả.

🡪parsing là chuyển đổi JSON (hoặc file JSON) thành đối tượng python

Dùng json.loads() để đọc từ chuỗi JSON:

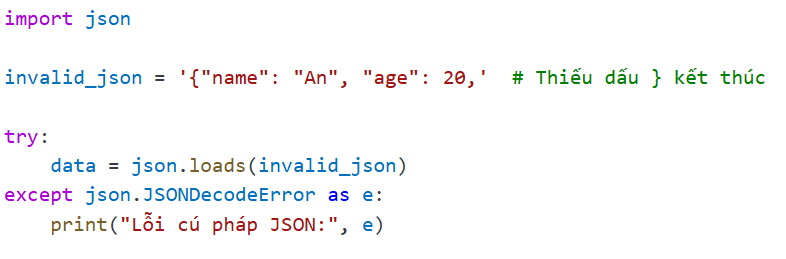


Dùng json.load() để đọc từ file:



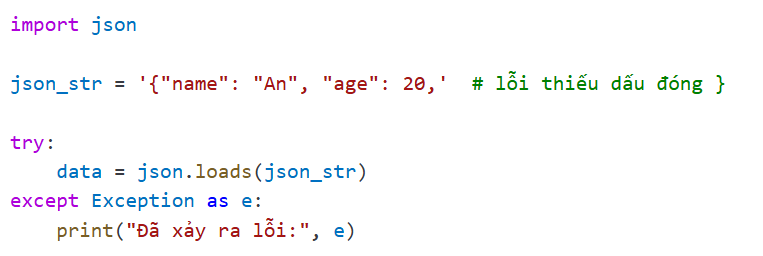
**Lưu ý:**

* Các **key trong JSON luôn là chuỗi**, nên nếu bạn dùng key số (1) trong Python thì sau khi load sẽ thành "1".
* tuple sẽ bị chuyển thành list, vì JSON không có tuple.
* Khi chuỗi JSON **không đúng định dạng**, Python sẽ báo lỗi. Khi đó ta nên dùng try ...except json.JSONDecodeError để bắt lỗi JSON chính xác.



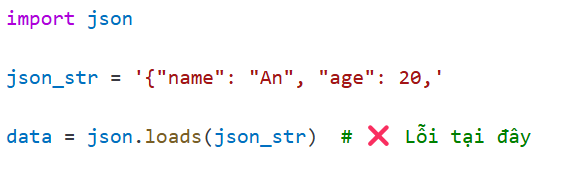
**Nếu không:**

* + Trường hợp 1: Có try...except, nhưng không chỉ rõ JSONDecodeError



➡️ Kết quả:

* Python **vẫn bắt được lỗi**, vì JSONDecodeError là một dạng của ValueError, và ValueError là một dạng của Exception.
* Nhưng bạn **không biết rõ** lỗi là lỗi JSON hay lỗi gì khác (trừ khi tự phân tích e).
  + Trường hợp 2: Không có try...except gì cả

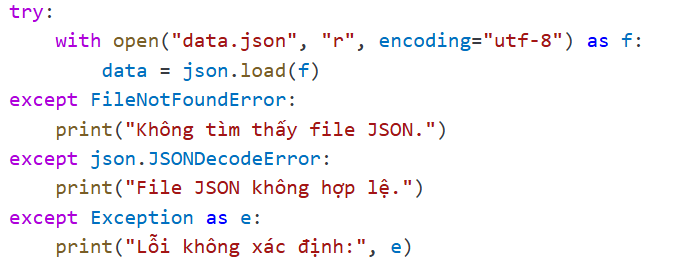


➡️ Kết quả:

* Chương trình **bị dừng (crash)** với lỗi:

json.decoder.JSONDecodeError: Expecting property name enclosed in double quotes: line 1 column 28 (char 27)

**Cách dùng an toàn nhất là:**



✅ Điều này giúp:

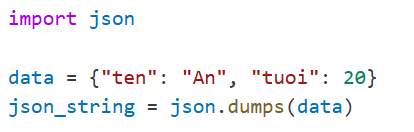
* Tránh chương trình bị crash.
* Dễ hiển thị thông báo lỗi rõ ràng cho người dùng.

1. **Encode JSON (Python 🡪JSON)**

Khi muốn **lưu dữ liệu vào file (đặc biệt là file .json)** để sử dụng lại sau này, bạn **phải chuyển dữ liệu Python thành chuỗi JSON**, vì file chỉ ghi được dữ liệu ở dạng chuỗi văn bản.

🡪Endcoding là qua trình chuyển đối tượng Python (như dict, list,...) thành chuỗi JSON

Dùng json.dumps() để chuyển đổi thành chuỗi JSON:



Dùng json.dump() để ghi JSON vào file:



Lưu ý:

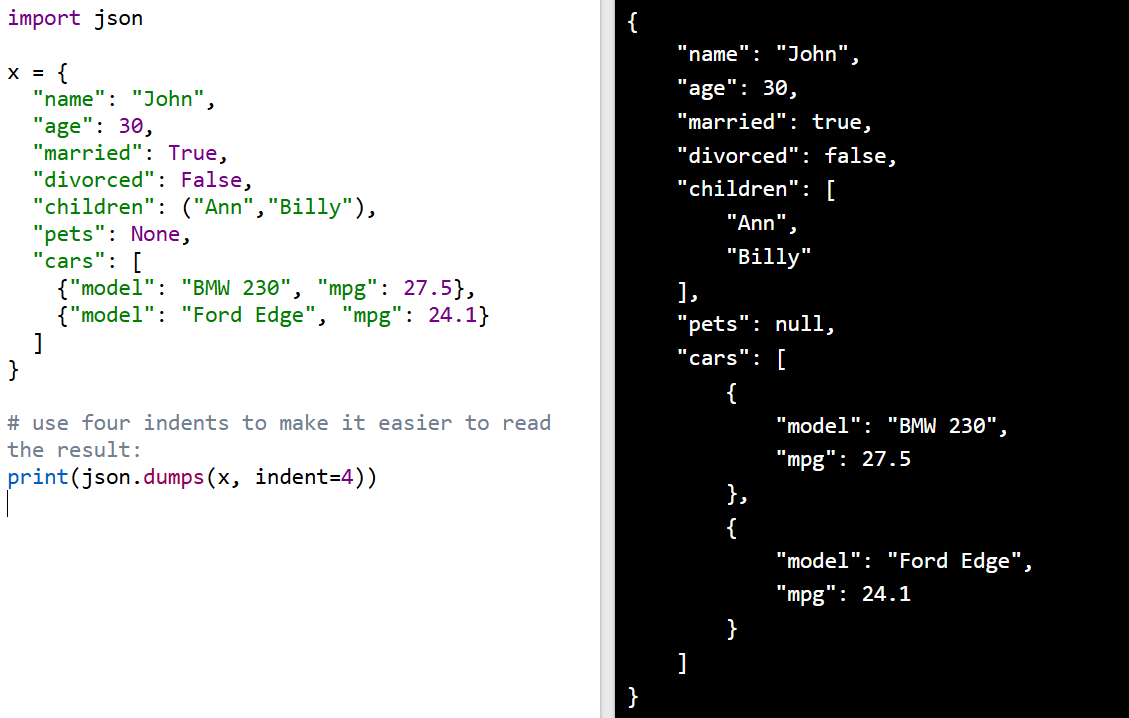
* + Khi bạn chuyển đổi từ Python sang JSON, các đối tượng Python sẽ được chuyển đổi thành JSON tương đương:

|  |  |  |
| --- | --- | --- |
| Python | JSON | Ví dụ |
| Dict | Object | Kết quả: {"name": "John", "age": 30, "married": true, "divorced": false, "children": ["Ann","Billy"], "pets": null, "cars": [{"model": "BMW 230", "mpg": 27.5}, {"model": "Ford Edge", "mpg": 24.1}]} |
| List, tuple | Array |
| Str | String |
| Int,float | Number |
| True | True |
| False | False |
| None | None |

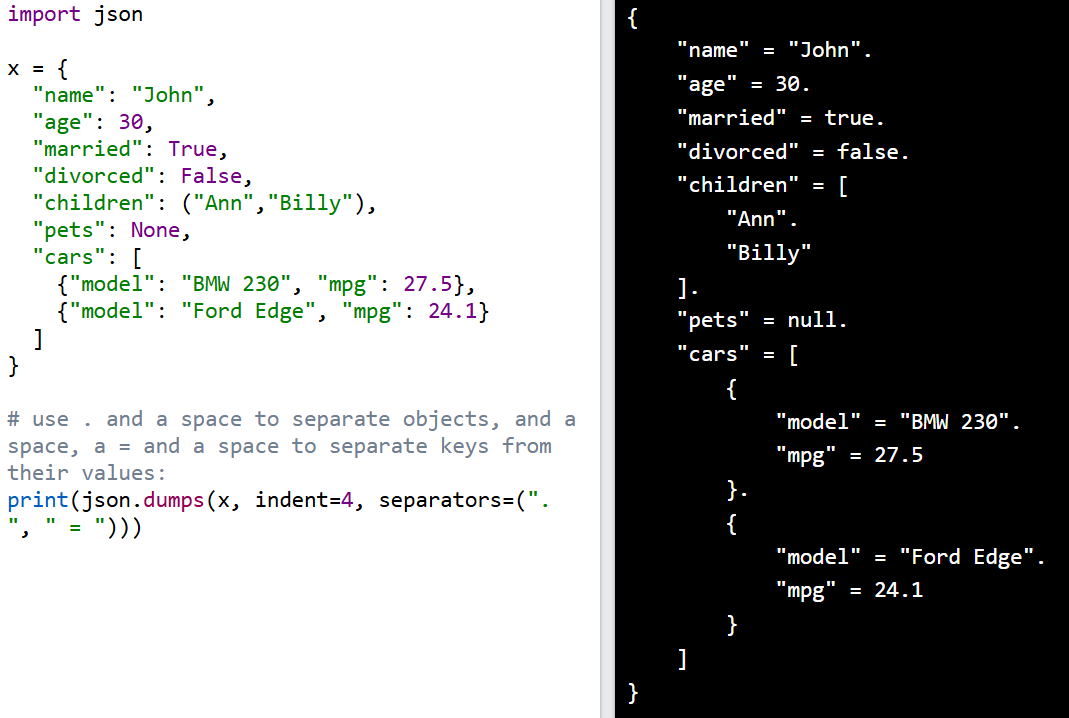
* + Không thể encoding : hàm (function), tuple làm key, object không tuần tự hóa được. Nếu cố encoding tuple làm key sẽ bị lỗi TypeError. Có thể dùng skipkeys=True để bỏ qua key không hợp lệ.

Định dạng kết quả : Ví dụ trên in ra một chuỗi JSON, nhưng không dễ đọc vì không có thụt lề và ngắt dòng.Phương pháp này json.dumps()có các tham số giúp đọc kết quả dễ dàng hơn:

**Thêm thụt lề (indent):**



**Tùy chỉnh dấu phân cách (separators):** Bạn cũng có thể định nghĩa các dấu phân cách, giá trị mặc định là (", ", ": "), nghĩa là sử dụng dấu phẩy và dấu cách để phân tách từng đối tượng, và dấu hai chấm và dấu cách để phân tách khóa với giá trị:



Sắp xếp key (sort\_keys): Sử dụng sort\_keystham số để chỉ định kết quả có được sắp xếp hay không

