Basic JSON Interview Questions & Answers in Python

Q1. What is JSON?

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is used to exchange da

Q2. How do you handle JSON in Python?

Python provides the 'json' module to encode and decode JSON data.

Example:
```
import json
data = {"name": "Alice", "age": 25}
json_string = json.dumps(data)
parsed = json.loads(json_string)
```

Q3. What’s the difference between json.dump() and json.dumps()?

- json.dump() → Writes JSON data directly to a file.
- json.dumps() → Converts Python object into JSON string.

Q4. What’s the difference between json.load() and json.loads()?

- json.load() → Reads JSON from a file and converts it to Python object.
- json.loads() → Parses JSON from a string and converts it to Python object.

Q5. How do you pretty-print JSON in Python?

Use indent parameter in json.dumps().

Example:
```
import json
data = {"name": "Alice", "age": 25, "city": "London"}
print(json.dumps(data, indent=4))
```

Q6. What Python data types are supported by JSON?

Python → JSON equivalents:

dict	object
list, tuple	array
str	string
int, float	number
True	true
False	false
None	null

Q7. Can JSON handle custom Python objects?

By default, no. JSON supports only basic types. For custom objects, use 'default' parameter in json.

Example:
```
import json
from datetime import datetime

def custom_serializer(obj):
    if isinstance(obj, datetime):
        return obj.isoformat()
    raise TypeError("Type not serializable")

print(json.dumps({"time": datetime.now()}, default=custom_serializer))
```