Paso 1: Importar la Librería JSON

Para manejar JSON en , necesitas importar la librería estándar json:

```
import json
```

Paso 2: Leer un Archivo JSON Local

Si tienes un archivo JSON en tu computadora, puedes leerlo con la función open() y luego usar json.load() para convertir el contenido en un objeto de .

```
with open('datos.json', encoding='utf-8') as archivo:
  datos = json.load(archivo)
```

- **Descripción**:
    - open('datos.json', encoding='utf-8'): Abre el archivo en modo lectura y asegura que se lea correctamente con caracteres especiales (tildes, ñ, etc.).
    - json.load(archivo): Convierte el contenido del archivo JSON en un objeto (puede ser una lista o un diccionario, dependiendo del JSON).

Paso 3: Leer un JSON desde una URL

Si tu archivo JSON está en línea, puedes usar las librerías requests y json para obtener y leer el archivo.

```
import requests


url = 'https://api.ejemplo.com/datos.json'

respuesta = requests.get(url)

datos = respuesta.json()  # Convierte el JSON en un objeto
```

- **Descripción**:
    - requests.get(url): Hace una solicitud HTTP para obtener el contenido del archivo JSON.
    - respuesta.json(): Convierte la respuesta en un objeto de Python.

Paso 4: Exploración Básica de los Datos

Supongamos que el contenido de datos.json es similar a este ejemplo:

json

```
{
 "personas": [
  {
   "nombre": "Juan",
   "edad": 30,
   "ciudad": "San José",
   "hobbies": ["leer", "jugar fútbol", "cocinar"]
  },
  {
   "nombre": "Ana",
   "edad": 25,
   "ciudad": "Cartago",
   "hobbies": ["escribir", "viajar"]
  }
 ]
}
```

Después de cargar el archivo JSON, puedes verificar el contenido de datos imprimiéndolo o inspeccionándolo en un entorno interactivo.

```
print(datos)  # Muestra el contenido completo
```

Paso 5: Acceder a los Datos del JSON

Una vez que tienes el archivo cargado en datos, puedes acceder a sus elementos usando índices y claves, dependiendo de la estructura del JSON.

Ejemplo de acceso paso a paso:

```
# Accedemos al primer nivel: clave "personas"
personas = datos["personas"]

# Accedemos al primer elemento de la lista "personas"
primera_persona = personas[0]

# Accedemos a los atributos de la primera persona
nombre = primera_persona["nombre"]
edad = primera_persona["edad"]
ciudad = primera_persona["ciudad"]
hobbies = primera_persona["hobbies"]
```

Paso 6: Trabajar con Listas dentro del JSON

En este caso, datos["personas"] es una lista de personas. Puedes recorrerla usando un bucle para obtener información sobre cada persona.

```
for persona in datos["personas"]:
    print(f"Nombre: {persona['nombre']}")
    print(f"Edad: {persona['edad']}")
    print(f"Ciudad: {persona['ciudad']}")
    print("Hobbies:", ", ".join(persona["hobbies"]))
    print("-----")
```

Paso 7: Acceder a Variables Anidadas

A veces, el JSON tiene estructuras más profundas, como diccionarios dentro de listas o listas dentro de diccionarios. Puedes acceder a estos datos con una combinación de índices y claves.

```
# Supongamos que queremos acceder al segundo hobby de la primera persona
segundo_hobby = datos["personas"][0]["hobbies"][1]
print(f"El segundo hobby de {datos['personas'][0]['nombre']} es: {segundo_hobby}")
```

Paso 8: Modificar Datos en el JSON

Una vez que el JSON está cargado como un objeto de , puedes modificar sus valores igual que harías con cualquier lista o diccionario.

```
# Cambiamos la edad de la primera persona
datos["personas"][0]["edad"] = 31

# Agregamos un nuevo hobby a la segunda persona
datos["personas"][1]["hobbies"].append("fotografía")

# Revisamos los cambios
print(datos)
```

Paso 9: Agregar Nuevos Elementos

Puedes agregar nuevos elementos al JSON como lo harías con cualquier lista o diccionario en .

```
# Agregar una nueva persona
nueva_persona = {
  "nombre": "Carlos",
  "edad": 40,
  "ciudad": "Heredia",
  "hobbies": ["correr", "ajedrez"]
}
datos["personas"].append(nueva_persona)


print(datos)
```

Paso 10: Guardar los Cambios en un Archivo JSON

Si has realizado cambios en los datos y quieres guardarlos, usa json.dump() para escribir los datos de vuelta a un archivo JSON.

```
with open('datos_actualizados.json', 'w', encoding='utf-8') as archivo:
  json.dump(datos, archivo, ensure_ascii=False, indent=4)
```

- **Descripción**:
  - 'w': Abre el archivo en modo escritura.
  - ensure_ascii=False: Asegura que los caracteres especiales se mantengan en UTF-8 (importante para tildes y ñ).
  - indent=4: Añade una indentación de 4 espacios para que el archivo sea más fácil de leer.