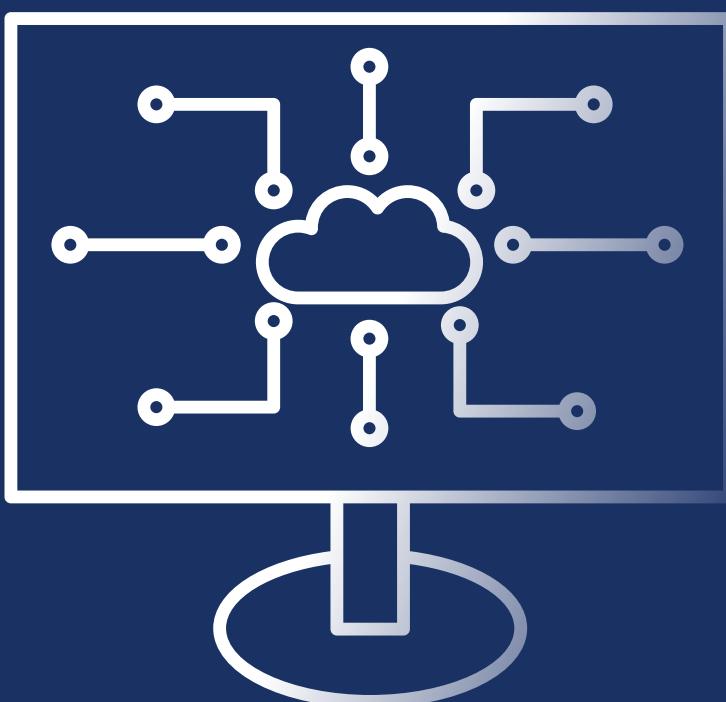


Programación I



Lic. Julia Monasterio
marmonasterio@uade.com.ar



Unidades

-  **Unidad 1:** Repaso de matrices, estructura, operaciones y uso en funciones
-  **Unidad 2:** Implementación y gestión en proyectos en Git
-  **Unidad 3:** Listas avanzadas, cadena de caracteres y expresiones regulares.
-  **Unidad 4:** Manipulación avanzada de diccionarios, tuplas y conjuntos.
-  **Unidad 5:** Excepciones y pruebas unitarias
-  **Unidad 6:** Archivos
-  **Unidad 7:** Recursividad

Clase N° 11

TEMAS

- Archivos JSON

Introducción

Archivos JSON - JAVASCRIPT OBJECT NOTATION

- Es una notación para la transferencia de datos que sigue un estándar específico
- Se utiliza comúnmente para representar información estructurada en formato texto, facilitando la transferencia de datos entre sistemas de manera eficiente.
- **Es similar a los diccionarios en Python, ya que los datos en JSON se organizan en una colección de pares clave:valor**

Librería JSON

- La biblioteca JSON permite leer y escribir archivos JSON.
- Permite convertir datos entre el formato JSON y las estructuras de datos de Python como **diccionarios**

Métodos para leer información

- **json.load():** lee el contenido de un archivo JSON y lo convierte en un diccionario de Python.
- **json.loads(cadena):** lee una cadena JSON y la convierte en un diccionario o lista de Python. Es útil cuando recibimos el json en formato **string**

Ejemplo 1

Ejemplo 1

- Leer la información de países de un archivo .json y mostrarlo por pantalla

Ejemplo 1

```
[  
  {  
    "id": 1,  
    "nombre": "Argentina"  
  },  
  {  
    "id": 2,  
    "nombre": "Brasil"  
  },  
  {  
    "id": 3,  
    "nombre": "Uruguay"  
  },  
  {  
    "id": 4,  
    "nombre": "Paraguay"  
  },  
  {  
    "id": 5,  
    "nombre": "Colombia"  
  }]
```

Ejemplo 1

```
import json

try:
    contenido = open("ejemplo1.json","rt")
    paises = json.load(contenido)
    print(paises)
    for pais in paises:
        print(pais["id"], "->", pais["nombre"])
except IOError as e:
    print("No se puede abrir el archivo",e)
finally:
    contenido.close()
```

Ejemplo 2

Ejemplo 2

- Calcular el promedio de las notas de alumnos almacenados en un json

Ejemplo 2

```
[{"nombre": "Juan", "notas": [8, 7, 9]},  
 {"nombre": "Pedro", "notas": [10, 6, 7]},  
 {"nombre": "Lourdes", "notas": [5, 7, 8]}]
```

Ejemplo 2

```
import json

try:
    archivo=open("alumnos.json", "rt")
    linea = archivo.readline()
    while linea:
        linea = linea.strip()
        if linea:
            estudiante = json.loads(linea)
            nombre = estudiante["nombre"]
            notas = estudiante["notas"]
            promedio = sum(notas) / len(notas)
            print(f"{nombre} tiene un promedio de {promedio:.2f}")
        linea = archivo.readline()

except FileNotFoundError:
    print("El archivo no se encuentra.")
except json.JSONDecodeError as e:
    print("Error al decodificar el JSON:", e)
finally:
    archivo.close()
```

Métodos para escribir información

- **json.dump(objeto, archivo, indent=4):** convierte un objeto de Python en una cadena de texto en formato JSON y la escribe directamente en un archivo.
 - **objeto:** el objeto de Python que queremos convertir a JSON (puede ser diccionario, lista, etc.)
 - **archivo:** un archivo abierto en modo escritura “w” o “a” donde se va a escribir el json
 - **indent (opcional):** numero de espacios para la indentacion del JSON para mejorar la legibilidad.

Métodos para escribir información

- **json.dumps(objeto, indent=4):** convierte un objeto de Python en una cadena de texto en formato JSON. No escribe directamente en un archivo, sino que genera una cadena que podemos almacenar en una variable
 - **objeto:** el objeto de Python que queremos convertir a JSON (puede ser diccionario, lista, etc.)
 - **indent (opcional):** numero de espacios para la indentacion del JSON para mejorar la legibilidad.

Ejemplo 3

Ejemplo 3

- Agregar un nuevo pais a la lista de diccionario de paises

Ejemplo 3

```
import json
try:
    archivo= open("ejemplo1.json","rt")

    lineas = archivo.read()
    print(lineas)

    paises = json.loads(lineas)

    paises.append({"id":6, "nombre":"Bolivia"})

    archivo.close()

    paisesJson = json.dumps(paises,indent=4)

    archivo_escritura= open("ejemplo1.json","wt")

    archivo_escritura.write(paisesJson)

except IOError:
    print("Error para encontrar archivo")
finally:
    archivo.close()
    archivo_escritura.close()
```

Ejemplo 4

- Agregar un nuevo pais a la lista de diccionario de paises con dump

Ejemplo 4

```
import json
|
try:
    archivo= open("ejemplo1.json","at")

    pais={"id":7,"nombre":"Israel"}
    paisesJson = json.dump(pais,archivo,indent=4)

    archivo.close()

except IOError:
    print("Error para encontrar archivo")
finally:
    archivo.close()
```

Ejercicio 1

Ejercicio 1

- Crear una lista de tareas que el usuario ingrese, persistirlas en un json y luego mostrarlas

Ejercicio 1

- Funcion cargar tareas

```
import json

def cargarTareas():
    tareas=[]
    bandera=True
    contador=1
    while bandera:

        tareaNombre = input("Ingrese la descripcion de la tarea o presione Enter para finalizar \n")

        if tareaNombre=="":
            bandera=False
        else:
            tarea={}
            tarea["id"]=contador
            tarea["descripcion"]=tareaNombre
            tareas.append(tarea)
            contador+=1

    return tareas
```

Ejercicio 1

- Función persistir tareas

```
def persistir_tareas(tareas,nombreArchivo):  
    try:  
        archivo=open(nombreArchivo, 'wt')  
        json.dump(tareas,archivo,indent=4)  
        print("Archivo generado correctamente!")  
    except IOError:  
        print("Error en generacion de archivo")  
    finally:  
        archivo.close()
```

Ejercicio 1

- Funcion mostrar tareas

```
def leerTareas(nombre_archivo):  
    try:  
        archivo = open(nombre_archivo, "r")  
        tareas = json.load(archivo)  
        print("Tareas que debe realizar")  
        for tarea in tareas:  
            print(f"ID - {tarea["id"]} - Descripcion: {tarea["descripcion"]}")  
    except FileNotFoundError:  
        print(f"No se encontró el archivo {nombre_archivo}")  
    except json.JSONDecodeError:  
        print("Error al decodificar el JSON.")
```

Ejercicio 1

- Programa Principal

```
#Programa Principal

nombre_archivo=input("Ingrese el nombre del archivo que desea crear\n")

persistir_tareas(cargarTareas(),nombre_archivo)

leerTareas(nombre_archivo)
```

Uso de with

- La utilización de with en la apertura de archivos permite gestionar automáticamente la apertura y el cierre del archivo.
- Cuando el bloque de código dentro del with se completa, el archivo se cierra automáticamente, incluso si ocurre una excepción dentro del bloque
- **Sintaxis:**

with open(nombre_archivo,modo_acceso) as nombre_archivo:

bloque

Ejercicio 2

Ejercicio 2

- Crear un programa que permita registrar ventas de productos, por cada venta registrar un id, un nombre de producto y una cantidad

Ejercicio 2 - Funcion cargar venta

```
import json
def cargarVentas():
    venta={}
    producto=input("Ingrese el producto que vendio\n")
    venta["producto"]=producto
    banderaPrecio=True
    while banderaPrecio:
        try:
            venta["precio"]= float(input("Ingrese el precio\n"))
            banderaPrecio=False
        except ValueError:
            print("Precio invalido. Por favor, ingrese un precio valido.")

    return venta
```

Ejercicio 2 - Funcion persistir venta

```
def persistirVentas(nombre_archivo,venta):
    try:
        with open(nombre_archivo,"rt") as archivo:
            ventas=json.load(archivo)
    except (FileNotFoundException, json.JSONDecodeError):
        print("Error")
        ventas=[]

    ventas.append(venta)

    try:
        with open(nombre_archivo,"wt") as archivo:
            json.dump(ventas,archivo,indent=4)
        print("Venta registrada")
    except IOError:
        print("Error de generacion de archivo.")
```

Ejercicio 2 - Funcion mostrar venta

```
def mostrarVentas(nombreArchivo):
    try:
        with open(nombreArchivo, "rt") as archivo:
            contenido= archivo.read()
            if contenido:
                ventas= json.loads(contenido)

                for venta in ventas:
                    print(f"Producto: {venta["producto"]} \t Precio: {venta["precio"]}")

            else:
                print("No hay ventas registradas")

    except (FileNotFoundException,json.JSONDecodeError):
        print("Archivo no existente")
```

Ejercicio 2 - Programa Principal

```
#Programa Principal

bandera=True
while bandera:
    try:
        opcion=int(input("----Bienvenidos a nuestro ECOMMERCE-----\n1 - Registrar venta\n 2 - Visualizar las ventas realizadas\
                          \n 3 - Salir\n"))
        if opcion==3:
            bandera=False
        elif opcion==1:
            persistirVentas("ventas.json",cargarVentas())
        elif opcion==2:
            mostrarVentas("ventas.json")
        else:
            print("Opcion invalida. Reinten")
    except ValueError:
        print("Opcion invalida. Reintente")
```

Resumen de la clase

- Archivos JSON: concepto
- Lectura de archivos JSON
- Escritura de archivos JSON
- Ejercicios

Muchas gracias!

Consultas?

