

# Trabajo Integrador

```
0     response = requests.get(url) # load from the website
1
2     # checking response.status_code (if you get 502, try rerunning the code)
3     if response.status_code != 200:
4         print(f"Status: {response.status_code} - Try rerunning the code")
5     else:
6         print(f"Status: {response.status_code}\n")
7
8     # using BeautifulSoup to parse the response object
9     soup = BeautifulSoup(response.content, "html.parser")
10
11    # finding Post images in the soup
12    images = soup.find_all("img", attrs={"alt": "Post image"})
13
14    # downloading images
15    i = 0
16    for image in images:
17        image.download(f"post_{i}.jpg")
18
19    print(f"Downloaded {len(images)} images")
```

Integrantes: Facundo Deseff y Matias Vargas

---

Materia: Programación I

**Profesora:** Cinthia Rigoni

## Introducción:

El objetivo principal de este proyecto es consolidar y aplicar de manera práctica los conceptos fundamentales adquiridos durante la cursada. Esto incluye el manejo de estructuras de datos esenciales como listas y diccionarios, la modularización del código a través de funciones , y la lectura de información desde archivos externos en formato CSV.

La aplicación desarrollada permite al usuario interactuar con un conjunto de datos de países, ofreciendo funcionalidades clave como la búsqueda de información , la aplicación de filtros por continente, población o superficie , el ordenamiento de los resultados y la generación de estadísticas relevantes. A través de este desarrollo, se busca afianzar las competencias técnicas en el procesamiento y la manipulación de datos en Python.

## Marco Teórico:

### 1. Listas

**Definición:** Una lista en Python es una estructura de datos que nos permite almacenar una colección ordenada y mutable de elementos. Se definen usando corchetes [ ].

**Aplicación en el TPI:** En nuestro proyecto, la estructura de datos central es una lista. La variable `países` (en la función `main`) es una lista que contiene todos los datos.

Funciones como `cargar_datos_api` y `cargar_datos_csv` crean y devuelven una `lista_paises`. Además, las funciones de filtro, como `filtrar_paises` y `buscar_pais_por_nombre`, crean una lista `resultados = []` para guardar los países que coinciden con el criterio de búsqueda.

### 2. Diccionarios

**Definición:** Un diccionario en Python es una estructura de datos que almacena elementos en pares clave: valor. Son colecciones mutables que permiten acceder a los valores de forma muy rápida a través de sus claves. Se definen usando llaves {}.

**Aplicación en el TPI:** Usamos un diccionario para representar cada país individual. En la función `cargar_datos_api`, creamos manualmente un diccionario país con las claves 'nombre', 'población', 'superficie' y 'continente'.

De forma similar, cargar\_datos\_csv usa csv.DictReader, que lee cada fila del archivo y la convierte automáticamente en un diccionario, lo cual es perfecto para nuestra estructura de datos.

### 3. Funciones

**Definición:** Una función es un bloque de código organizado y reutilizable que realiza una tarea específica. Ayudan a modularizar el código, hacerlo mas legible y evitar la repetición. Se definen en Python usando la palabra clave def.

**Aplicación en el TPI:** Para cumplir con la consigna de código modular, todo el programa esta dividido en funciones. Cada función tiene una única responsabilidad. Por ejemplo:

- *cargar\_datos\_csv()* : Solo se encarga de leer el archivo.
- *mostrar\_menu()* : Solo se encarga de imprimir las opciones.
- *validar\_num()* : Es una función auxiliar que solo valida un ingreso numérico.
- *main()* : Es la función principal que coordina la ejecución de todas las demás.

### 4. Condicionales

**Definición:** Las estructuras condicionales (principalmente if, elif y else) permiten que un programa ejecute diferentes bloques de código según si se cumple o no una condición booleana (Verdadero o Falso).

**Aplicación en el TPI:** Los condicionales son la base de la lógica del programa. Los usamos por ejemplo en:

- *main()* : Para ejecutar una acción según la opción elegida por el usuario (Ej: *if opcion == "1"*).
- *filtrar\_paises()* : Para validar el rango (Ej: *if min\_poblacion > max\_poblacion*).
- *validar\_num()* : Para asegurar que el numero no sea negativo (Ej: *if numero < 0*).

## 5. Ordenamientos

**Definición:** El ordenamiento es el proceso de organizar los elementos de una colección (como una lista) en un orden específico (ascendente o descendente).

**Aplicación en el TPI:** "Implementamos esta funcionalidad en la función ordenar\_paises(), la cual muestra un sub-menu para elegir el criterio. Para ordenar la lista de diccionarios, usamos la función sorted() de Python. El parámetro key de esta función fue fundamental: lo usamos con una función lambda (ej: key=lambda x: x['nombre']) para indicarle a sorted() qué clave del diccionario debía usar para comparar y ordenar."

## 6. Estadísticas Básicas

**Definición:** Las estadísticas básicas son cálculos que resumen un conjunto de datos. Esto incluye operaciones como calcular promedios (media aritmética), encontrar valores máximos y mínimos, y contar frecuencias.

**Aplicación en el TPI:** "Creamos la función mostrar\_estadisticas(paises). Esta función recorre la lista para calcular los datos.

- Usamos max() y min() (junto con una key y lambda) para encontrar el país con mayor y menor población/superficie.
- Usamos sum() y len() para calcular los promedios globales.
- Finalmente, usamos un diccionario auxiliar para agrupar los países por su clave 'continente' y así poder calcular las estadísticas de cada uno."

## 7. Archivos CSV

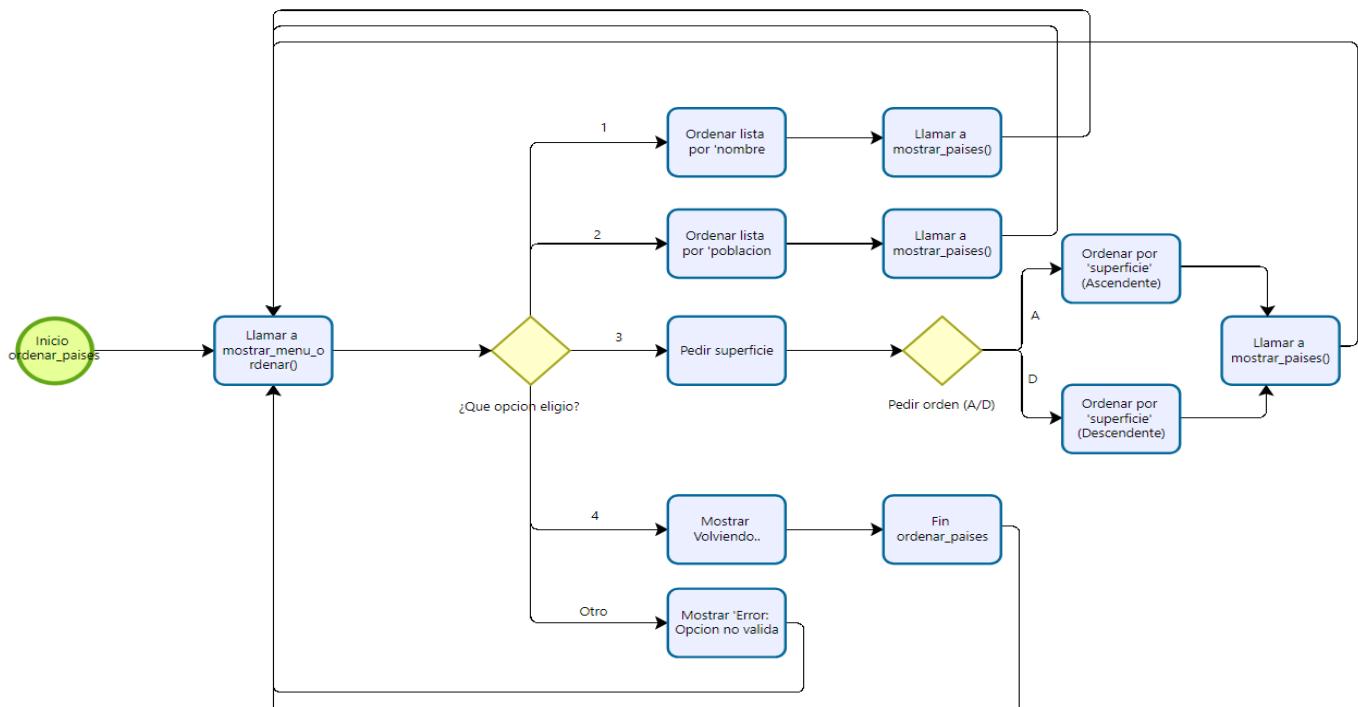
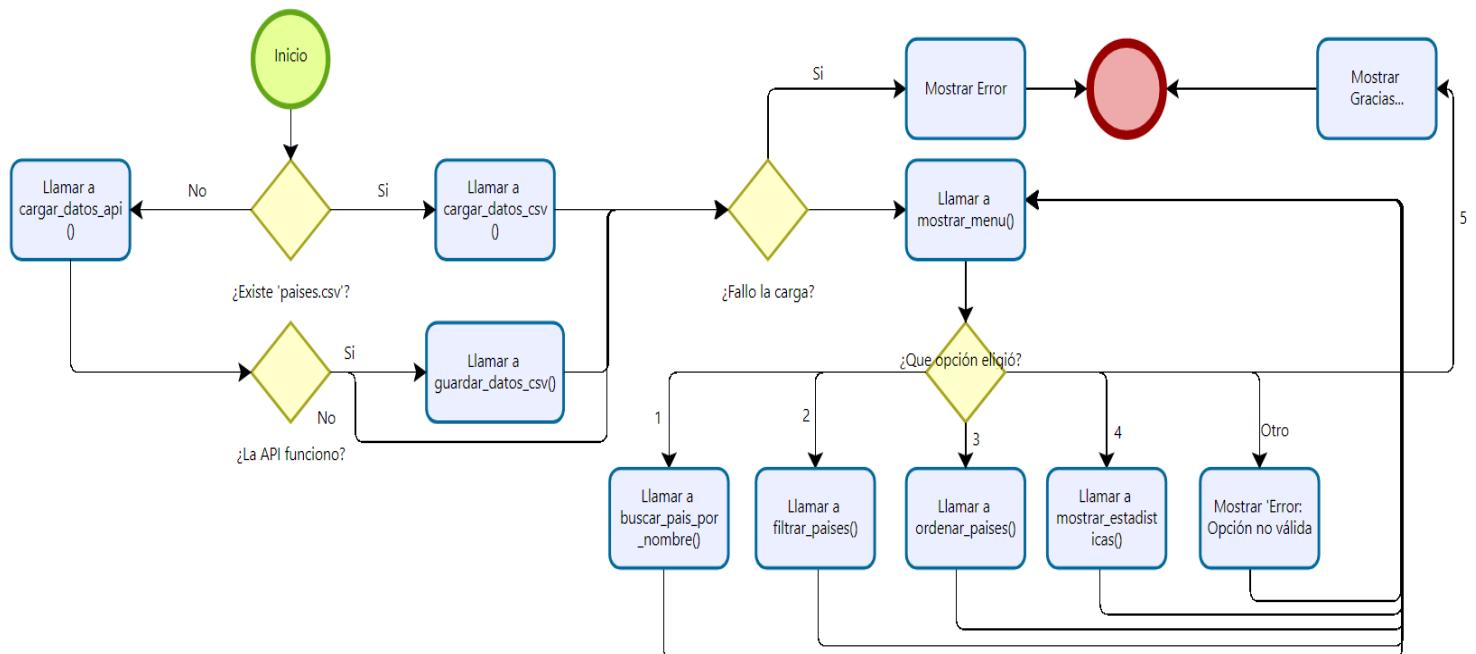
**Definición:** Un archivo CSV (Valores Separados por Comas) es un formato de archivo de texto plano que se usa para almacenar datos tabulares (como una hoja de cálculo). Cada línea del archivo representa una fila, y las columnas se separan por un delimitador (generalmente una coma).

**Aplicación en el TPI:** Usamos el módulo csv de Python para manejar la persistencia de los datos.

- guardar\_datos\_csv(): Usa csv.DictWriter para escribir nuestra lista de diccionarios en el archivo paises.csv.
- cargar\_datos\_csv(): Usa csv.DictReader para leer ese archivo y volver a convertirlo en una lista de diccionarios.

Esto nos permite (como se ve en `main()`) no tener que consultar la API cada vez que se ejecuta el programa, solo la primera vez.

# Diagrama de Flujo



### **Prueba del código:**

```
--- Menú Principal: Gestión de Países ---
1. Buscar un país por nombre
2. Filtrar países
3. Ordenar países
4. Mostrar estadísticas
5. Salir
6. Actualizar datos desde la API
Seleccione una opción (1-6):
```

Menu principal

### **Opcion filtrar paises:**

```
Seleccione una opción (1-6): 2

SubMenu: Filtrar Países
1. Filtrar por Continente
2. Filtrar por Rango de Población
3. Filtrar por Rango de Superficie
4. Volver al menú principal
Seleccione una opción de filtro (1-4): 2
--- Filtro por Población ---
Ingresar la población MÍNIMA: 450000000
Ingresar la población MÁXIMA: 8000000000

--- 35 Países Encontrados ---
- Kenya
  Población: 53,330,978
  Superficie: 580,367 km2
  Continente: Africa
-----
- México
  Población: 130,575,786
  Superficie: 1,964,375 km2
  Continente: North America
-----
- Tanzania
  Población: 68,153,004
  Superficie: 947,303 km2
  Continente: Africa
-----
- Nigeria
  Población: 223,800,000
  Superficie: 923,768 km2
  Continente: Africa
```

Opcion mostrar estadísticas:

Seleccione una opción (1-6): 4

Total de países: 250

País más poblado: India (1,417,492,000 hab.)

País menos poblado: British Indian Ocean Territory (0 hab.)

País más grande: Russia (17,098,246 km<sup>2</sup>)

País más chico: Vatican City (0 km<sup>2</sup>)

Promedio de población: 32,077,982 hab.

Promedio de superficie: 603,453 km<sup>2</sup>

Asia:

Países: 50

Promedio población: 92,548,727 hab.

Promedio superficie: 623,367 km<sup>2</sup>

Oceania:

Países: 27

Promedio población: 1,831,430 hab.

Promedio superficie: 315,868 km<sup>2</sup>

Africa:

Países: 58

Promedio población: 25,214,904 hab.

Promedio superficie: 522,768 km<sup>2</sup>

North America:

Países: 41

Promedio población: 14,768,994 hab.

Promedio superficie: 595,048 km<sup>2</sup>

Europe:

Países: 55

Promedio población: 15,228,447 hab.

Promedio superficie: 436,362 km<sup>2</sup>

Antarctica:

Países: 5

Promedio población: 340 hab.

Promedio superficie: 2,802,422 km<sup>2</sup>

South America:

Países: 14

## **Conclusiones:**

La realización de este Trabajo Práctico Integrador nos ha permitido poner en práctica el conjunto de herramientas fundamentales de Programación 1, aplicándolas a un problema concreto de gestión y análisis de datos.

1. **Sobre el Desarrollo:** El desarrollo de la aplicación para gestionar datos de países requirió la implementación de un menú interactivo que permitiera al usuario navegar por las distintas funcionalidades. El núcleo del proyecto fue la correcta lectura del archivo CSV y el almacenamiento de los datos en una estructura de listas y diccionarios, lo cual facilitó enormemente la manipulación de la información.
2. **Sobre los Desafíos:** El desafío más significativo fue la implementación de las funciones de filtrado y ordenamiento. Lograr que el programa pudiera filtrar por rangos (como población o superficie) y a la vez ordenar los resultados de manera ascendente o descendente requirió una planificación cuidadosa de la lógica. También, asegurar la modularidad del código, dividiendo cada responsabilidad (buscar, filtrar, calcular estadísticas) en funciones separadas, fue una actividad que resultó clave para mantener el código limpio y mantenible.
3. **Sobre el Aprendizaje:** Este proyecto consolidó nuestra comprensión sobre la importancia de las estructuras de datos: mientras la lista sirvió como contenedor principal, los diccionarios fueron perfectos para representar cada país con sus atributos (Nombre, Población, etc.).
4. **Sobre el Trabajo Grupal:** La coordinación como equipo fue fundamental. Ya que nos dividimos para poder realizar el trabajo de forma más fluida y organizada, además este tipo de actividad es buena para ir viviendo la experiencia de armar un proyecto con alguien más, ya que es muy distinto encargarse solo de un proyecto que acompañado, esto hace que estas actividades además de que sean realizadas con un mejor desempeño, hace que se disfrute el proceso.
5. **Cierre:** En conclusión, este trabajo nos deja una base sólida en la manipulación de datos con Python. Cumplimos con el objetivo de afianzar el uso de estructuras y modularización, y el proyecto final es un reflejo tangible de las competencias adquiridas en la materia.

Bibliografia:

**Python Software Foundation.** (2025). *The Python Standard Library – Módulo csv (comma-separated value files)*. Documentación Oficial de Python 3.10. Recuperado el 2 de noviembre de 2025, de:

<https://docs.python.org/3/library/csv.html>

**Python Software Foundation.** (2025). *The Python Tutorial – Data Structures (Listas, Tuplas, Conjuntos, Diccionarios)*. Documentación Oficial. Recuperado el 1 de noviembre de 2025, de:

<https://docs.python.org/3/tutorial/datastructures.html>