Trabajo integrador final Programación

Gestión de Datos de Países en Python: filtros, ordenamientos y estadísticas

Alumnos:

Liendo Mateo, Avila Lucas y Pagano Amanda

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.

Programación

Docente Titular

Rigoni Cinthia

02 de Noviembre de 2025

Tabla de contenido

Trabajo Práctico integrador

Introducción

El presente informe describe el desarrollo de un programa en Python cuyo objetivo es gestionar información sobre países.

El sistema permite obtener datos desde una API externa, almacenarlos en archivos CSV y realizar distintas operaciones como búsquedas, filtrados, agregar o eliminar, ordenamientos y cálculos estadísticos básicos.

A través de este proyecto se aplicaron diversos conceptos fundamentales de la programación estructurada en Python, como **listas, diccionarios, funciones, condicionales, ordenamientos, estadísticas básicas y manejo de archivos CSV**.

También se trabajó con la modularización del código para mejorar la organización, manteniendo funciones separadas por responsabilidades específicas.

Objetivos:

Objetivo general:

Desarrollar un programa en Python capaz de gestionar, analizar y almacenar información de países utilizando estructuras de datos y conceptos básicos de programación estructurada.

Objetivos específicos:

- Aplicar los conceptos de **listas y diccionarios** para almacenar información de manera ordenada y eficiente.
- Implementar **funciones** que organicen el código y eviten repeticiones.
- Utilizar **condicionales** para controlar el flujo del programa y validar las opciones del usuario.
- Realizar **ordenamientos** y **filtrados** de datos según distintos criterios.
- Calcular **estadísticas básicas** (promedios, valores máximos y mínimos) sobre la población y superficie de los países.
- Agregar o eliminar datos (países) del archivo CSV.
- Practicar el manejo de **archivos CSV** para guardar y recuperar información.
- Integrar todos los conceptos anteriores dentro de un programa modular y funcional que facilite la comprensión de los datos obtenidos desde una API externa.

Conceptos teóricos aplicados:

Listas

Las listas son estructuras que permiten almacenar varios elementos en una sola variable.

Se utilizan para guardar los datos de los países (por ejemplo, nombre, población, superficie, continente, etc.).

Son estructuras **dinámicas y ordenadas**, lo que facilita recorrerlas, modificarlas o filtrarlas.

En el proyecto, las listas se usan para guardar los registros leídos del archivo CSV o cargados desde la API.

Ejemplo aplicado:

La lista países contiene varios diccionarios, uno por cada país.

Diccionarios

Los diccionarios permiten guardar pares **clave: valor**.

Son útiles para representar objetos con atributos, como un país con sus datos (nombre, capital, región, población, superficie, etc.).

En el código, cada país se maneja como un diccionario dentro de una lista general.

Ejemplo aplicado:

```
pais = {"nombre": "Argentina", "capital": "Buenos Aires", "poblacion": 45000000,
```

De esta forma se accede fácilmente a cada dato usando su clave.

Funciones

Las funciones permiten dividir el programa en partes reutilizables. Cada función cumple una tarea específica, lo que facilita la lectura, depuración y mantenimiento del código.

En el proyecto se definieron funciones para:

- Buscar países por nombre.
- Filtrar por continente.
- Ordenar por población o superficie.
- Calcular estadísticas.
- Cargar o guardar datos desde/ hacia un archivo CSV.

El uso de funciones también posibilitó la **modularización**, es decir, distribuir el código en distintos archivos (módulos), importando las funciones necesarias desde cada uno.

Condicionales

Las estructuras condicionales (if, elif, else) permiten tomar decisiones según distintas situaciones.

Se emplearon para validar opciones del menú, verificar la existencia de países y ejecutar acciones según la selección del usuario.

Ejemplo aplicado:

```
if opcion == 1:
    mostrar_paises(paises)
elif opcion == 2:
    filtrar_por_continente(paises)
else:
    print("Opción no válida")
```

Ordenamientos

El programa implementa distintos tipos de ordenamiento utilizando funciones de Python como sorted() o el método sort(), junto con **funciones lambda** o claves específicas.

En nuestro código, **sorted** y la función **lambda** trabajan juntas para ordenar la lista de diccionarios paises basándose en el valor de la clave "superficie".

La función sorted() es la encargada de realizar la acción de ordenamiento sobre una colección iterable (en este caso, la lista paises).

- **Creación de una Nueva Lista:** A diferencia del método .sort(), la función sorted() **no modifica la lista original** (paises). En su lugar, crea y retorna una **nueva lista** con los elementos ordenados, la cual se asigna a la variable ordenados.

- **Argumentos Usados:**
    - **Primer argumento (paises):** La lista de elementos a ordenar.
    - **key=lambda p: p["superficie"]:** Le dice a sorted() qué valor debe usar para comparar los elementos. Sin esta clave, sorted() intentaría comparar los diccionarios completos, lo que causaría un error o daría un resultado sin sentido.
    - **reverse=reverse:** Controla la dirección del orden. La variable reverse toma el valor True (si el usuario eligió 'D' de descendente) o False (si eligió 'A' de ascendente).

La función lambda actúa como una **función de clave** que le dice a sorted() *cómo* extraer el valor numérico para la comparación de cada elemento.

- **Definición:** lambda p: p["superficie"]
- **Mecánica:**
    - sorted() toma un país (un diccionario) de la lista y lo pasa a la función lambda (el argumento p).
    - lambda **retorna** el valor asociado a la clave "superficie" de ese diccionario.
    - sorted() usa este valor numérico (la superficie) para comparar el país actual con los demás países.

Esto permite organizar la información según la población, superficie o nombre del país.

Ejemplo aplicado:

```
paises_ordenados = sorted(paises, key=lambda p: p["poblacion"], reverse=True)
```

Estadísticas básicas

Se aplicaron operaciones simples de análisis de datos, como obtener valores máximos, mínimos y promedios de la población o superficie de los países. Estas operaciones se realizaron mediante el uso de funciones como max(), min() y sum() combinadas con comprensión de listas.

Ejemplo aplicado:

```
promedio = sum(p["poblacion"] for p in paises) / len(paises)
```

Archivos CSV

Los archivos CSV (Comma Separated Values) se utilizaron para almacenar los datos obtenidos de la API o cargados manualmente. Permiten guardar la información en un formato estructurado y fácil de leer. Se emplearon funciones del módulo csv de Python para **leer** y **escribir** los archivos.

Ejemplo aplicado:

```python
import csv

with open("paises.csv", "w", newline="", encoding="utf-8") as archivo:
    escritor = csv.DictWriter(archivo, fieldnames=paises[0].keys())
    escritor.writeheader()
    escritor.writerows(paises)
```

Aplicación práctica de los conceptos:

En el proyecto se desarrolló un menú principal que permite al usuario seleccionar distintas operaciones. Cada opción llama a funciones específicas definidas en módulos independientes, lo que mejora la estructura general del código.

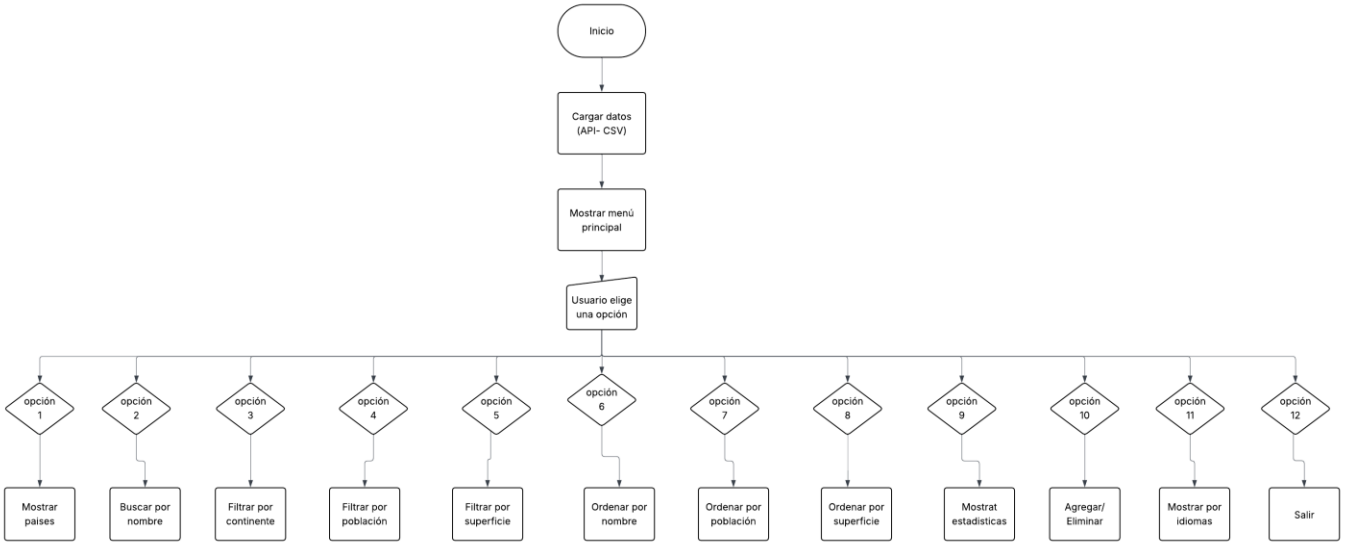
Ejemplos de funciones principales:

- **Opción 1:** Mostrar todos los países.
- **Opción 2:** Buscar un país por nombre.
- **Opción 3:** Filtrar por continente.
- **Opción 4:** Calcular estadísticas (promedio de población o superficie).
- **Opción 5:** Ordenar países.
- **Opción 6–10:** Otras operaciones de gestión, carga, eliminación y guardado de datos.

Los datos se obtienen inicialmente desde una **API REST**, se procesan y se guardan en un archivo **CSV** para su uso local posterior.

Flujo de operaciones principales

El siguiente diagrama ilustra el funcionamiento general del programa, mostrando el proceso desde la carga inicial de datos hasta la ejecución de las distintas opciones disponibles en el menú principal. Cada decisión representa una acción posible del usuario dentro del sistema.



Imágenes salidas por consola:

Opción 9:

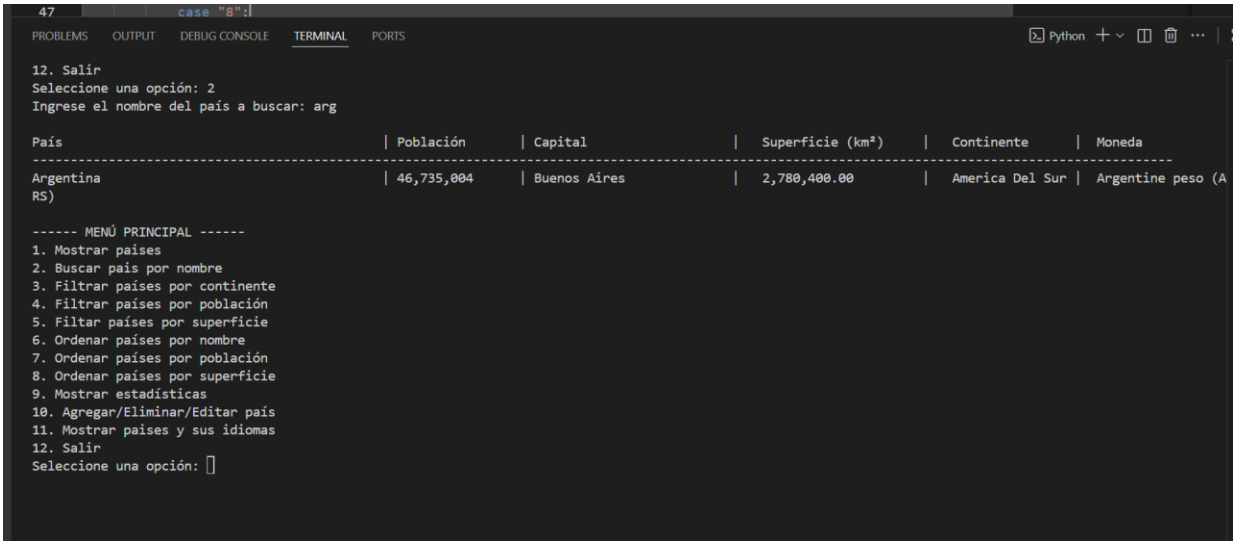
```
hp@DESKTOP-53SCNNN MINGW64 ~/OneDrive/Escritorio/trabajos_practicos/Integrador-Programacion (main)
$ C:/Users/hp/AppData/Local/Programs/Python/Python313/python.exe c:/Users/hp/OneDrive/Escritorio/tr
practicos/Integrador-Programacion/integradorLimpio.py
11. Salir
Seleccione una opción: 9

°°°°°°°°°°°°°°°°°°°°Estadísticas Generales de Países°°°°°°°°°°°°°°°°°°°°
-------------------------------------------------------------------------
Indicador                          |                    Valor
-------------------------------------------------------------------------
Cantidad total de países           |                      248
-------------------------------------------------------------------------
Población total                    |            7,751,341,877
-------------------------------------------------------------------------
Población promedio                 |            31,255,410.79
-------------------------------------------------------------------------
Superficie total (km²)             |           149,823,726.66
-------------------------------------------------------------------------
Superficie promedio (km²)          |               604,127.93
-------------------------------------------------------------------------
País con mayor población           | China
-------------------------------------------------------------------------
→ Habitantes                       |            1,402,112,000
-------------------------------------------------------------------------
País con menor población           | Heard Island and McDonald Islands
-------------------------------------------------------------------------
→ Habitantes                       |                        0
-------------------------------------------------------------------------
País con mayor superficie          | Russia
-------------------------------------------------------------------------
→ Superficie (km²)                 |            17,098,242.00
-------------------------------------------------------------------------
País con menor superficie          | Vatican City
-------------------------------------------------------------------------
→ Superficie (km²)                 |                    0.44
-------------------------------------------------------------------------
```

La opción 9 muestra las estadísticas del CSV :

- Cantidad de paises

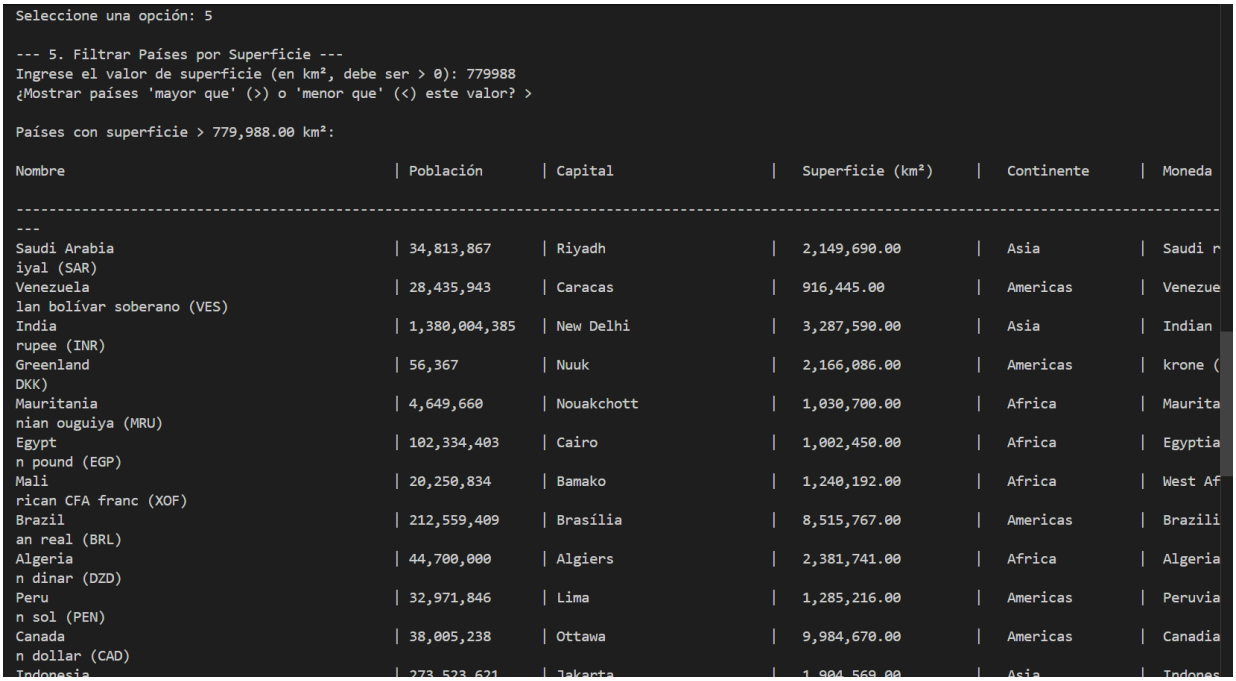
- Población total
- Población promedio
- Superficie total
- Superficie promedio
- País con mayor población
- País con menor población
- País con mayor superficie
- País con menor superficie

Opción 2:



La opción 2 busca el país, ya sea por nombre o aproximación ej. Argentina (arg).

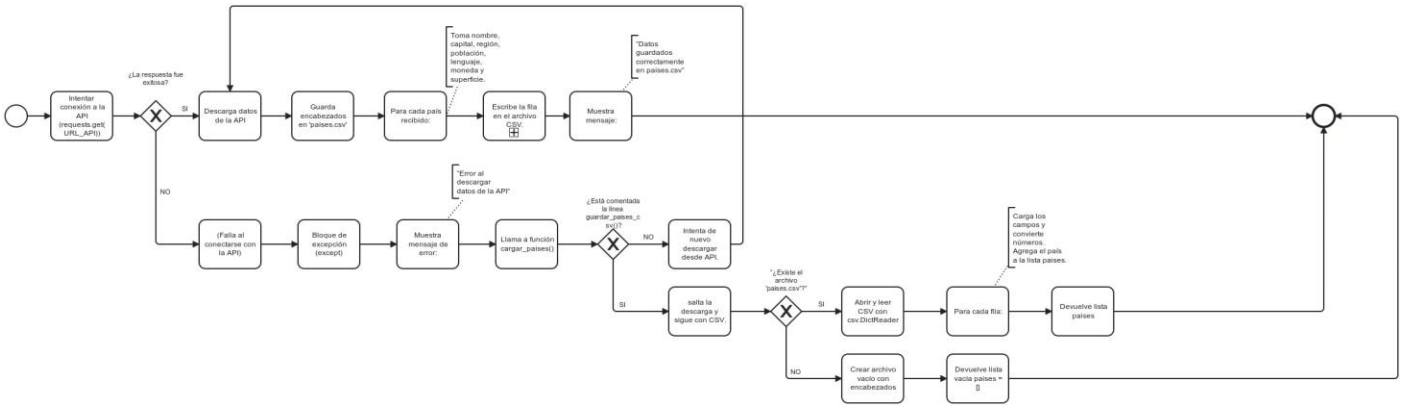
Opción 5:



```
Seleccione una opción: 5

--- 5. Filtrar Países por Superficie ---
Ingrese el valor de superficie (en km², debe ser > 0): 779988
¿Mostrar países 'mayor que' (>) o 'menor que' (<) este valor? >

Países con superficie > 779,988.00 km²:

Nombre                                  | Población       | Capital              |   Superficie (km²)    |   Continente      |  Moneda

-------------------------------------------------------------------------------------------------------------------------------------------
---
Saudi Arabia                            | 34,813,867      | Riyadh               |   2,149,690.00        |   Asia            |  Saudi r
iyal (SAR)
Venezuela                               | 28,435,943      | Caracas              |   916,445.00          |   Americas        |  Venezue
lan bolívar soberano (VES)
India                                   | 1,380,004,385   | New Delhi            |   3,287,590.00        |   Asia            |  Indian
rupee (INR)
Greenland                               | 56,367          | Nuuk                 |   2,166,086.00        |   Americas        |  krone (
DKK)
Mauritania                              | 4,649,660       | Nouakchott           |   1,030,700.00        |   Africa          |  Maurita
nian ouguiya (MRU)
Egypt                                   | 102,334,403     | Cairo                |   1,002,450.00        |   Africa          |  Egyptia
n pound (EGP)
Mali                                    | 20,250,834      | Bamako               |   1,240,192.00        |   Africa          |  West Af
rican CFA franc (XOF)
Brazil                                  | 212,559,409     | Brasília             |   8,515,767.00        |   Americas        |  Brazili
an real (BRL)
Algeria                                 | 44,700,000      | Algiers              |   2,381,741.00        |   Africa          |  Algeria
n dinar (DZD)
Peru                                    | 32,971,846      | Lima                 |   1,285,216.00        |   Americas        |  Peruvia
n sol (PEN)
Canada                                  | 38,005,238      | Ottawa               |   9,984,670.00        |   Americas        |  Canadia
n dollar (CAD)
Indonesia                               | 273,523,621     | Jakarta              |   1,904,569.00        |   Asia            |  Indones
```

La opción 5 pide que ingrese un valor numérico de superficie y pregunta si desea filtrar países con superficie con menor o mayor numero del que ingreso, mostrando según la elección solo los que cumplan la condición.

Diagrama de flujo de nuestras funciones principales (cargar_paises y guardar_paises_csv):



La función descarga los datos de países desde la API y los guarda en un archivo CSV. Si la API no responde, carga los datos desde el CSV existente o crea uno nuevo vacío para seguir funcionando sin conexión.

Conclusiones:

A lo largo del desarrollo del trabajo práctico se integraron los principales conceptos de la programación estructurada en Python. Se aplicaron estructuras de datos (listas y diccionarios), condicionales, funciones, manejo de archivos CSV y cálculo de estadísticas básicas, logrando un sistema funcional y bien modularizado.

La experiencia permitió comprender la importancia de dividir el código en módulos y funciones, así como el valor de mantener un flujo de operaciones claro y organizado.

El programa permite **consultar, filtrar, ordenar y analizar información de países**, obtenida desde una API y almacenada en un archivo CSV. Además, se implementaron funciones adicionales para **mostrar el lenguaje y la moneda de cada país**, ampliando el alcance del análisis y aportando información complementaria al usuario.

Referencias

https://chatgpt.com/ (Correcciones ortográficas)

(Lucidchart diagrama flujo funcionamiento básico programa) - https://lucid.app/lucidchart/6cdabff1-c199-426a-8b0f-bde599d41b07/edit?viewport_loc=-2211%2C-797%2C3068%2C1556%2C0_0&invitationId=inv_940a07fe-c08c-439c-a211-271c39f391a1 Requests (**librería de Python** que sirve para **hacer solicitudes HTTP)**

API Rest Countries - REST Countries

BPMN.io Diagrama funciones principales (https://demo.bpmn.io/new)

Tecnicatura Universitaria en Programación.