UNIVERSIDAD TECNOLOGICA NACIONAL

Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.

Programación 1

Trabajo Práctico Integrador

Estudiantes

- Domenicale Doré, Maria Luz
- Egea Ruiz, Magaly

Docentes

- Titular: Cinthia Rigoni
- Tutora: Ana Mutti

LINK AL VIDEO: click aquí 

LINK AL PROGRAMA: click aquí 

11/11/2025

Índice

Consignas Trabajo Práctico Integrador

🗒️ Trabajo Práctico Integrador (TPI)

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

🎯 Objetivo

Desarrollar una aplicación en Python que permita gestionar información sobre países, aplicando listas, diccionarios, funciones, estructuras condicionales y repetitivas, ordenamientos y estadísticas. El sistema debe ser capaz de leer datos desde un archivo CSV, realizar consultas y generar indicadores clave a partir del dataset.

⚙️ Consignas generales

- Lenguaje: Python 3.x
- Estructuras: listas, diccionarios, funciones
- Archivos: lectura desde CSV
- Código claro, comentado y modularizado (una función = una responsabilidad)
- Validaciones de entradas y manejo básico de errores
- Trabajo en equipos de 2 personas

🌍 Dominio (dataset de países)

Cada país estará representado con los siguientes datos:

- Nombre (string)
- Población (int)
- Superficie en km² (int)
- Continente (string)
- Ejemplo de registro CSV:

Código

nombre,poblacion,superficie,continente

Argentina,45376763,2780400,América

Japón,125800000,377975,Asia

Brasil,213993437,8515767,América

Alemania,83149300,357022,Europa

🧠 Requerimientos técnicos

1. Diseño (previo al código)

Informe teórico explicando los conceptos aplicados:

- Listas
- Diccionarios
- Funciones
- Condicionales
- Ordenamientos
- Estadísticas básicas
- Archivos CSV

Introducción

Tener muchos datos puede ser abrumador y no servir de nada si no se manejan bien. Hoy en día, los datos vienen de muchos lugares y formatos diferentes, y las empresas necesitan usarlos de manera efectiva para aprovecharlos al máximo. Con las herramientas adecuadas, los datos se pueden convertir en un activo muy valioso, porque ayudan a entender mejor a los clientes, hacer predicciones más precisas y mejorar procesos o servicios. También permiten crear nuevos modelos de negocio, como servicios basados en inteligencia artificial, que necesitan datos de buena calidad para funcionar bien.

"*Estar basado en datos significa usar datos, cualquiera que sea el método, para manejar la complejidad del estado, el almacenamiento, el acceso, la calidad y el contexto para permitir que las organizaciones hagan realidad sus aspiraciones basadas en datos, fundamentales para el éxito del negocio digital.*"

Gartner "Introducción a las soluciones de gestión de datos para 2023, Adam Ronthal, Ehtisham Zaidi, 14 de febrero de 2023

La **Gestión de Datos** es básicamente todo lo que tiene que ver con juntar, organizar, guardar y usar información de una manera ordenada, segura y que no sea tan costosa.

La idea principal es que, gracias a esa información organizada, las personas o empresas puedan tomar mejores decisiones y hacer cosas que les convengan más.
Por ejemplo: una tienda online puede analizar los datos de sus ventas para ver qué productos se venden más y así decidir cuáles conviene volver a comprar o promocionar.

Esto tiene que respetar ciertas normas, como la privacidad, para proteger los datos de las personas.
Tener una buena forma de manejar los datos es súper importante, porque hoy en día muchas empresas ya no dependen tanto de máquinas o cosas físicas, sino de la información que tienen y cómo la usan para generar valor.

Los sistemas de gestión de datos son súper importantes porque las empresas manejan muchísima información que viene de distintos lugares. Entonces, necesitan una forma más práctica y ordenada de controlar todos esos datos, como si estuvieran dentro de un mismo sistema, aunque en realidad estén repartidos en muchos lados.

Estos sistemas se apoyan en diferentes cosas, como:

Bases de datos, donde se guarda información más organizada, tipo nombres, precios o clientes.

Lagos de datos, que son como grandes depósitos donde se guarda todo tipo de información, incluso la que todavía no está ordenada.

Almacenes de datos, que sirven para analizar la información y sacar conclusiones.

Aunque hoy existen programas que ayudan a automatizar muchas tareas, todavía hay muchas cosas que se hacen **a mano**, porque los sistemas son muy grandes y complicados. Y cuando algo se hace manualmente, hay más chances de cometer errores.

Por eso, el objetivo ahora es **lograr que todo se maneje más solo**, y ahí aparece la **base de datos autónoma**, que es una tecnología nueva capaz de administrarse casi sin ayuda humana.

Los **sistemas de gestión de macrodatos** se encargan de manejar muchísima información que llega todo el tiempo y en diferentes formatos, como los datos que se generan en redes sociales, cámaras o dispositivos conectados a Internet. Estos sistemas ayudan a juntar, organizar, guardar y analizar todos esos datos de manera segura y ordenada. Gracias a eso, las empresas pueden mejorar sus productos, anticiparse a problemas, dar una mejor atención a los clientes y trabajar de forma más eficiente. En pocas palabras, los macrodatos les dan a las empresas más oportunidades para aprender y tomar mejores decisiones.

Hoy en día, **manejar datos es más complicado** porque las empresas tienen información que crece muy rápido, viene de muchos lados diferentes y cambia todo el tiempo. No alcanza con solo juntar datos; hay que saber **qué datos hay, dónde están y cómo usarlos**, porque si no no sirven para nada. Además, con tanta información hay que mantenerla rápida y segura, cumplir con leyes de privacidad como el **RGPD** (una ley europea que protege los datos personales y da a las personas control sobre su información), y poder procesarla y convertirla en algo útil sin que sea un trabajo eterno.

Para esto, las empresas usan varias cosas: crear una **capa de descubrimiento** (una herramienta para buscar y entender qué datos tienen y cómo se pueden usar), tener **entornos de ciencia de datos** (espacios de trabajo donde los científicos de datos organizan, transforman y analizan información con herramientas que automatizan el trabajo pesado), usar **bases de**

datos autónomas (bases de datos que se ajustan y optimizan solas sin que alguien tenga que hacerlo manualmente) y **bases de datos convergentes** (bases que pueden manejar todos los tipos de datos modernos en un solo sistema). También es importante tener una **capa de consulta común** (un sistema que permite acceder a los datos aunque estén guardados en distintos lugares o formatos, sin tener que moverlos o transformarlos a mano).

Todo esto ayuda a las empresas a **analizar los datos más rápido y mejor**, sacar conclusiones útiles, tomar decisiones inteligentes y adelantarse a la competencia. En pocas palabras, **los datos hoy son un recurso muy valioso**, y si se manejan bien, se pueden aprovechar al máximo.

La razón de todo esto es que, si no se organizan y gestionan bien los datos, se pierde tiempo, dinero y oportunidades, y además se corre el riesgo de errores o de no cumplir con la ley. Por eso las mejores prácticas y las tecnologías modernas buscan **automatizar, unificar y agilizar** todo el proceso de gestión de datos.

Problema que resuelve el proyecto

En este trabajo se desarrolló una aplicación en Python destinada a la gestión de datos de países, incluyendo información sobre nombre, población, superficie y continente. El sistema permite cargar los datos desde un archivo CSV, asegurando que la información se almacene de manera uniforme y fácil de manipular. La aplicación ofrece funcionalidades como **agregar o actualizar registros, buscar países por nombre, aplicar filtros por continente o rangos de población y superficie, ordenar registros según distintos criterios y calcular estadísticas básicas**, como el país con mayor o menor población, promedios de población y superficie, y cantidad de países por continente.

De esta manera, el proyecto resuelve el problema de **analizar y organizar manualmente información extensa y diversa**, evitando errores y ahorrando tiempo. La herramienta facilita el acceso rápido a los datos, la comparación entre registros y la obtención de estadísticas confiables, proporcionando una solución práctica para estudiar, investigar y tomar decisiones basadas en información precisa y estructurada.

Objetivos del proyecto.

El objetivo principal de nuestro Trabajo Práctico Integrador (TPI) es **aplicar todo lo que aprendimos en Programación 1** en un proyecto concreto. La idea es que podamos usar y entender mejor las **estructuras de datos**, como listas y diccionarios, y ver cómo funcionan en la práctica para organizar y manejar información.

También queremos reforzar la **modularización del código**, creando funciones que hagan tareas específicas y que podamos usar varias veces en el programa, así nuestro código queda más claro y fácil de entender. Otro punto importante es aprender a **trabajar con archivos CSV**, leyendo y guardando información, y usar **condicionales y bucles** para procesar los datos de manera automática, sin tener que hacerlo todo a mano.

Además, buscamos poder **ordenar, filtrar y analizar los datos**, sacar estadísticas básicas y generar información útil que nos permita tomar decisiones más rápidas y confiables. Durante

el desarrollo del proyecto también aprendimos a **validar datos y manejar errores**, para que el programa funcione correctamente ante distintos tipos de entradas.

Por último, este trabajo nos permitió **trabajar en equipo**, coordinarnos entre la parte de programación y la documentación, y practicar la **lógica de programación** aplicada a un caso real, lo cual nos da experiencia y preparación para futuros proyectos en los que haya que combinar teoría, práctica y análisis de datos.

Qué evalúa la materia

En este TPI, la materia busca que podamos demostrar todo lo que aprendimos durante la cursada y aplicarlo en un proyecto concreto. Nos permite evaluar cómo usamos las estructuras de datos, como listas y diccionarios, para almacenar, organizar y procesar información de manera eficiente. También se observa nuestra capacidad de modularizar el código, creando funciones que cumplan tareas específicas y que faciliten la lectura y el mantenimiento del programa.

Además, se evalúa cómo manejamos archivos CSV, tanto en la lectura como en la escritura de datos, y cómo aplicamos condicionales y bucles para controlar el flujo del programa. Otro punto importante es la implementación de filtros, ordenamientos y estadísticas básicas, así como la forma en que validamos los datos y controlamos errores para que el sistema funcione correctamente en distintos escenarios.

Por último, la materia también valora nuestra capacidad de documentación y presentación, asegurando que el código esté bien comentado, que el proyecto tenga un README claro y que podamos explicar de manera efectiva cómo funciona la aplicación. Además, se toma en cuenta el trabajo en equipo, la coordinación entre los miembros, la organización de tareas y la comunicación constante, lo cual fortalece nuestras habilidades blandas, como la colaboración, la planificación y la resolución de problemas en conjunto, competencias fundamentales para proyectos reales.

Para organizar el proyecto y cubrir todas las áreas, nos dividimos las tareas de la siguiente manera: Magaly se encargó de la programación, las validaciones y el flujo del programa, y yo, Luz me ocupé del marco teórico, el PDF, el video y el README. Esta división nos permitió trabajar de manera coordinada y aprovechar nuestras fortalezas, al mismo tiempo que desarrollamos habilidades de planificación y comunicación en equipo.

Marco Teórico

En nuestro TPI aplicaremos conceptos fundamentales de Python para desarrollar una aplicación que gestione datos de países de manera organizada. Usaremos **listas y diccionarios** para almacenar la información, **funciones** para modularizar el código, **condicionales y bucles** para controlar el flujo y validar datos, **ordenamientos y estadísticas básicas** para analizar la información, y **archivos CSV** para guardar y leer los datos de forma persistente.

Este marco teórico nos permite entender cómo cada concepto se aplica en el proyecto, haciendo que la aplicación sea funcional, confiable y fácil de mantener, mientras trabajamos coordinadas en equipo.

Listas de Python

¿Qué son?

Las listas son una de las estructuras más usadas en Python. Sirven para guardar varios elementos juntos bajo un mismo nombre. Dentro de una lista se pueden mezclar distintos tipos de datos, como palabras, números o valores lógicos (verdadero o falso), todo dentro de una misma colección.

¿Cómo están organizadas?

Las listas son **ordenadas**, lo que significa que cada elemento tiene un lugar fijo dentro de la secuencia. El primer elemento se encuentra en la posición 0, el segundo en la 1, y así sucesivamente. También pueden contener elementos repetidos, ya que cada uno tiene su propio índice.

¿Se pueden modificar?

Sí, las listas son **modificables**, lo que permite cambiar, agregar o eliminar elementos después de haberlas creado. Es posible reemplazar valores, insertar nuevos en posiciones específicas o borrar los que ya no se necesiten. Además, se pueden combinar con otras listas o colecciones.

¿Cómo se accede a los elementos?

Para acceder a un elemento se usa su número de posición, llamado índice. También se puede contar desde el final usando índices negativos (por ejemplo, -1 para el último elemento). Si se necesita obtener varios valores a la vez, se puede indicar un rango de posiciones.

¿Para qué sirven?
Las listas son muy útiles cuando se trabaja con varios datos relacionados, ya que permiten recorrerlos fácilmente con bucles (*for* o *while*) y realizar operaciones sobre cada elemento. Gracias a su flexibilidad, se usan en todo tipo de programas, desde cálculos simples hasta manejo de grandes volúmenes de información.

En nuestro TPI, las listas se aplican directamente en la estructura catálogo principal: lista_paises, que es una lista de diccionarios, que almacena todos los países como elementos. Gracias a ellas podemos recorrer los registros con bucles, aplicar filtros y calcular estadísticas. Esta organización permitió que el programa sea flexible y que cada operación se realice sobre la misma colección de datos.

Diccionarios en Python

¿Qué son?
Los diccionarios en Python son un tipo de estructura que sirve para guardar información en forma de pares **clave–valor**. Cada "clave" funciona como un nombre o etiqueta que identifica a un "valor". Por ejemplo, podemos guardar datos de un país usando claves como "nombre", "población", "superficie" o "continente". Esto hace que los diccionarios sean muy útiles cuando queremos acceder rápidamente a información específica dentro de un conjunto de datos.

¿Cómo se escriben?
Se escriben entre **llaves** `{ }`, separando cada par con comas. Las claves y los valores se separan con dos puntos. Por ejemplo:
`{"nombre": "Argentina", "poblacion": 45376763, "superficie": 2780400, "continente": "América"}`.
En este caso, cada clave ("nombre", "poblacion", etc.) está asociada a un valor. Las claves deben ser únicas, es decir, no pueden repetirse, pero los valores pueden ser del tipo que sea: texto, número, lista, o incluso otro diccionario.

¿Se pueden modificar?
Sí, los diccionarios son **modificables**, lo que significa que se pueden cambiar, agregar o

eliminar pares de clave y valor en cualquier momento. Si se usa una clave que ya existe, su valor se reemplaza automáticamente por el nuevo. También se pueden agregar nuevas claves o quitar las que ya no sean necesarias.

¿Tienen un orden?
Desde Python 3.7, los diccionarios mantienen el orden en el que se agregaron los elementos, por lo que al recorrerlos se muestran en el mismo orden en que fueron creados.

¿Para qué sirven?
En el TPI, los diccionarios se pueden usar para representar datos complejos de una manera clara y ordenada. Por ejemplo, si queremos almacenar información sobre varios países, cada país podría ser un diccionario con sus propias claves y valores. Así, acceder a los datos es mucho más rápido y organizado.

En el proyecto, cada país se representa como un diccionario con claves como nombre, población, superficie en Km^2 y continente. Esto nos permitió acceder rápidamente a la información específica de cada país y mantener un formato uniforme en todo el catálogo. La combinación de listas y diccionarios fue clave para cumplir con la consigna de gestionar datos de manera clara y ordenada.

Funciones en Python

¿Qué son?
Las funciones son bloques de código que realizan una tarea específica y que podemos reutilizar cada vez que lo necesitemos. En lugar de repetir código una y otra vez, definimos la función una vez y luego la llamamos cada vez que haga falta.

¿Cómo se usan?
Primero se define la función, dándole un nombre y, opcionalmente, parámetros de entrada. Luego, en el programa, la "llamamos" cuando queremos que ejecute su tarea. Las funciones también pueden devolver un resultado para que lo usemos en otra parte del programa.

¿Por qué sirven en el TPI?
Las funciones pueden usarse para tareas comunes como: agregar un país a la base de datos, actualizar sus datos, filtrar por continente, ordenar por población, o calcular estadísticas (por

ejemplo “¿cuál es el país con mayor superficie?”). Esto hace que el código quede más limpio, organizado y fácil de mantener.

En nuestro código, cada función cumple una responsabilidad única: agregar_pais() gestiona altas, ordenar_paises() organiza los registros, y mostrar_estadisticas() calcula indicadores. Esta modularidad hizo que el programa sea más legible y fácil de mantener, y permitió que cada ajuste se realizara sin afectar al resto del sistema.

Condicionales en Python

¿Qué son?

Las condicionales son estructuras que permiten que el programa tome decisiones. En lugar de ejecutar siempre el mismo código, Python puede seguir distintos caminos según se cumpla o no una condición. Es decir, el programa “piensa”: si algo es verdadero, hace una cosa; si no, hace otra.

¿Cómo funcionan?

Para crear una condición se usa la palabra clave `if`, y opcionalmente se pueden agregar `elif` (si no, pero…) y `else` (en cualquier otro caso). Python evalúa una expresión lógica, y si el resultado es verdadero, ejecuta el bloque de código correspondiente.

¿Qué tipo de condiciones se pueden usar?

Las condiciones pueden comparar números, textos o valores lógicos, y también se pueden combinar con operadores como **and**, **or** y **not** para construir reglas más complejas. Por ejemplo, se puede verificar si un número es mayor que otro o si un texto coincide con una palabra específica.

¿Se pueden anidar?

Sí, se pueden poner condicionales dentro de otras. Esto sirve cuando se necesita tomar decisiones más detalladas, como validar distintos rangos o casos especiales.

¿Para qué sirven?

Las condicionales son esenciales para controlar el flujo del programa. En el TPI, pueden servir para validar los datos ingresados (por ejemplo, que la población no sea negativa), comprobar si un país existe antes de agregarlo o decidir qué acción realizar según la opción elegida en el menú.

En el TPI, las condicionales se usan para validar entradas y controlar el flujo del menú. Por ejemplo, verificamos que la población ingresada sea mayor que cero y que el país no exista antes de agregarlo. También se aplican en los filtros en cascada, donde cada condición decide si un país se incluye o no en los resultados.

Bucles o estructuras repetitivas en Python

¿Qué son?

Los bucles son estructuras que permiten repetir una o varias acciones varias veces sin tener que escribir el mismo código muchas veces. En otras palabras, sirven para automatizar tareas que se repiten, como recorrer una lista, procesar datos o mostrar información varias veces.

¿Cómo funcionan?

Python ofrece principalmente dos tipos de bucles: **for** y **while**.

El bucle **for** se usa cuando sabemos cuántas veces queremos repetir algo o cuando recorremos una colección (como una lista o un diccionario).

El bucle **while** se usa cuando queremos repetir una acción mientras se cumpla una condición. Cuando la condición deja de ser verdadera, el bucle termina.

¿Se pueden combinar con otras estructuras?

Sí, los bucles se pueden usar junto con condicionales para hacer programas más inteligentes. Por ejemplo, se puede recorrer una lista de países y usar un `if` dentro del bucle para filtrar solo los que cumplan cierta condición, como los que tengan una población mayor a cierto número.

¿Cómo se controlan?

Existen palabras clave que ayudan a manejar el comportamiento de los bucles:

- **break**: detiene el bucle antes de que termine.
- **continue**: salta una vuelta y pasa a la siguiente.
- **else**: se ejecuta cuando el bucle termina sin interrupciones.

¿Para qué sirven?

Los bucles son fundamentales para trabajar con conjuntos de datos. En el TPI, pueden servir para recorrer la lista de países, sumar sus poblaciones, calcular promedios, buscar uno en particular o aplicar filtros según un criterio. Gracias a los bucles, el código se vuelve más simple, ordenado y eficiente.

En nuestro programa, los bucles recorren la lista de países para realizar operaciones como navegar por el menú, sumar poblaciones, calcular promedios o aplicar filtros. También se combinan con condicionales para mostrar solo los países que cumplen ciertos criterios. Esto permitió automatizar tareas repetitivas y mantener el código simple y eficiente.

Ordenamientos en Python

¿Qué son?

El ordenamiento es una forma de organizar los datos según un criterio, ya sea alfabético, numérico, ascendente o descendente. Esto permite ver la información de manera más clara y compararla fácilmente.

¿Cómo funcionan?

En Python existen métodos que permiten ordenar listas o colecciones de forma rápida. El más común es **sort()**, que ordena directamente la lista, y **sorted()**, que crea una nueva lista ordenada sin modificar la original.

¿Se puede personalizar el orden?

Sí. Se puede elegir si el orden será ascendente o descendente, e incluso se puede definir una función personalizada para ordenar según un campo específico (por ejemplo, por población o por nombre de país).

¿Para qué sirven?

En el TPI, los ordenamientos son útiles para mostrar los países según su nombre, superficie o población, lo que facilita la interpretación de los datos. También ayudan a preparar la información para generar reportes o estadísticas.

En el TPI, los ordenamientos se implementaron con funciones como sorted() y criterios personalizados. Gracias a esto, pudimos mostrar los países ordenados por nombre, población, superficie o continente, en forma ascendente o descendente. Esta funcionalidad facilita la interpretación de los datos y cumple con los requerimientos técnicos.

Estadísticas básicas en Python

¿Qué son?

Las estadísticas básicas son cálculos que ayudan a resumir y entender mejor los datos. Permiten obtener valores importantes como promedios, máximos, mínimos o conteos de una colección.

¿Cómo se aplican en Python?

Se pueden realizar de varias formas: usando funciones integradas como **max()**, **min()**, **sum()** o **len()**, o bien a través de librerías especializadas como **statistics** o **numpy** para cálculos más complejos.

¿Qué información brindan?

Sirven para analizar los datos y encontrar patrones o valores representativos. Por ejemplo, calcular el país con mayor población, el promedio de superficie o cuántos países pertenecen a cada continente.

¿Para qué sirven?

En el TPI, permiten interpretar los datos de manera más práctica y generar conclusiones a partir de la información almacenada, lo cual es muy útil al momento de mostrar resultados o tomar decisiones.

En nuestro proyecto, las estadísticas se aplican para calcular el país con mayor y menor población, los promedios de población y superficie, y la cantidad de países por continente. Estas funciones permiten interpretar los datos de manera práctica y generar conclusiones confiables a partir del catálogo.

Archivos CSV en Python

¿Qué son?

Los archivos CSV (Comma-Separated Values) son archivos de texto donde los datos se separan por comas. Son muy usados porque permiten almacenar información en forma de tabla y se pueden abrir con programas como Excel o Google Sheets.

¿Cómo funcionan?

Cada línea del archivo representa una fila y cada valor separado por comas representa una columna. En Python, se pueden leer o escribir estos archivos fácilmente usando el módulo **csv** o librerías como **pandas**.

¿Se pueden modificar?

Sí, se pueden agregar, actualizar o eliminar datos, e incluso convertir la información de una lista o diccionario directamente a un archivo CSV. Según Python Software Foundation (2024), el módulo csv permite leer y escribir archivos de manera sencilla.

¿Para qué sirven?

En el TPI, los archivos CSV son útiles para guardar la información de los países de forma permanente. Permiten cargar los datos cuando se inicia el programa o actualizarlos sin tener que escribir todo de nuevo.

En el TPI, el archivo CSV es la base de persistencia de datos. Implementamos la función guardar_catalogo_en_csv() para centralizar la escritura y evitar errores. De esta manera, cada modificación en la lista de diccionarios se refleja en el archivo, asegurando que la información se mantenga actualizada y coherente con la consigna.

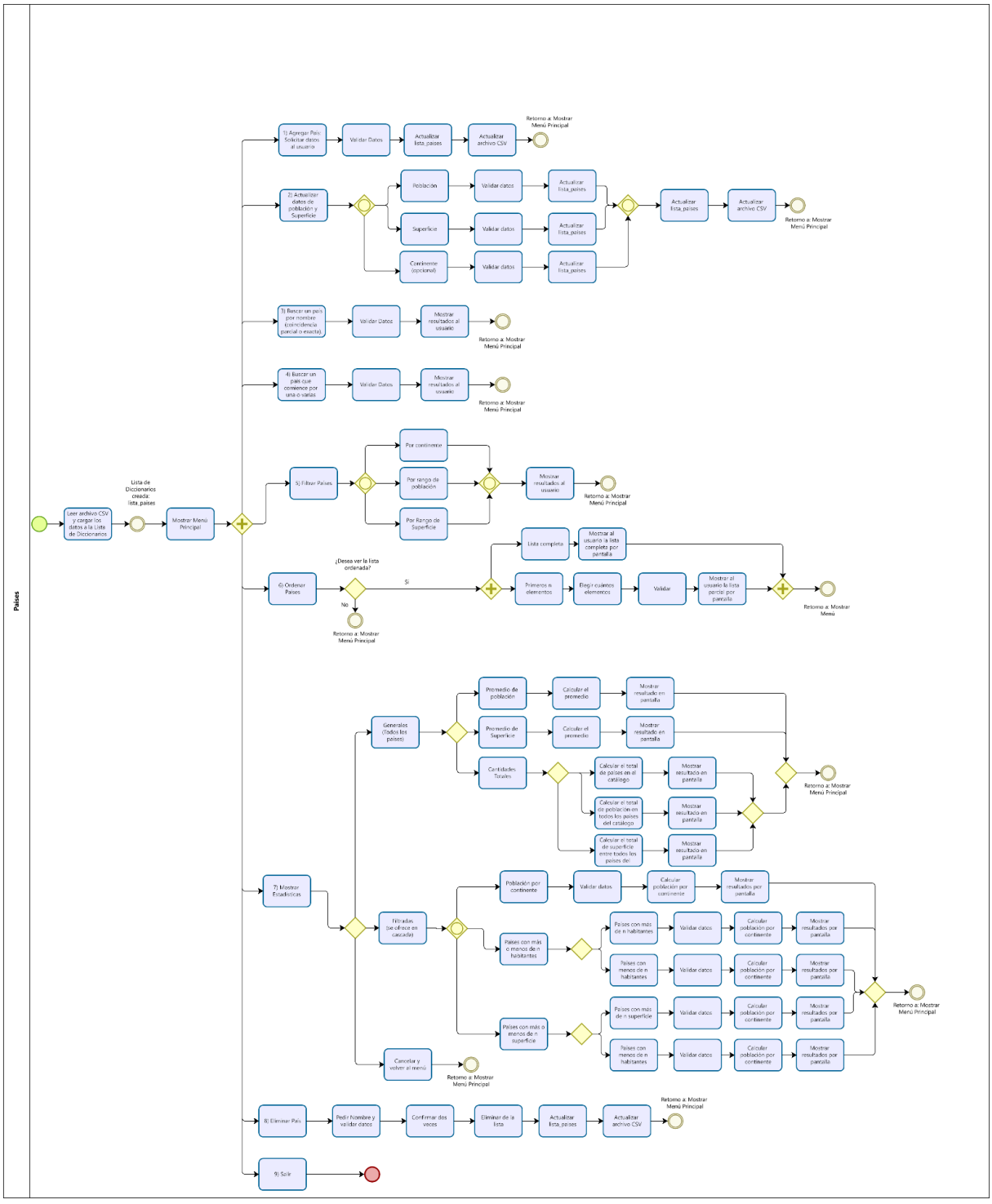
A través de este trabajo buscamos aplicar de manera práctica todos los conceptos aprendidos durante la cursada. Utilizamos estructuras de datos como **listas y diccionarios** para organizar la información, **funciones** para mantener un código ordenado y reutilizable, y **condicionales y bucles** para automatizar tareas y tomar decisiones dentro del programa. También incorporamos el uso de **archivos CSV** para guardar y recuperar los datos, y **estadísticas básicas** para analizar la información de forma más completa.

Todo esto nos permitió desarrollar una aplicación funcional, clara y modular, que demuestra el manejo integral de Python y la importancia de combinar teoría y práctica en un proyecto real.

Caso Práctico - Desarrollo del Programa

Flujo de operaciones principales:

- El **diagrama completo** puede visualizarse en la siguiente página, pero si se quiere en **alta resolución** se puede acceder desde este enlace al repositorio, lo que facilita su lectura y análisis: clic aquí  .
- El **código completo del programa** puede visualizarse aquí: clic aquí 


Países
Leer archivo CSV y cargar los datos a la Lista de Diccionarios
Lista de Diccionarios creada: lista_paises
Mostrar Menú Principal
1) Agregar País: Solicitar datos al usuario
Validar Datos
Actualizar lista_paises
Actualizar archivo CSV
Retorno a: Mostrar Menú Principal
2) Actualizar datos de población y Superficie
Población
Superficie
Continente (opcional)
Validar datos
Actualizar lista_paises
3) Buscar un país por nombre (coincidencia parcial o exacta)
Mostrar resultados al usuario
4) Buscar un país que comience por una o varias
5) Filtrar Países
Por continente
Por rango de población
Por Rango de Superficie
6) Ordenar Países
¿Desea ver la lista ordenada?
Si
No
Lista completa
Mostrar al usuario la lista completa por pantalla
Primeros n elementos
Elegir cuántos elementos
Validar
Mostrar al usuario la lista parcial por pantalla
Retorno a: Mostrar Menú
7) Mostrar Estadísticas
Generales (Todos los países)
Promedio de población
Promedio de Superficie
Cantidades Totales
Calcular el promedio
Mostrar resultado en pantalla
Calcular el total de países en el catálogo
Calcular el total de población en todos los países del catálogo
Calcular el total de superficie entre todos los países del
Mostrar resultado en pantalla
Filtrados (se ofrece en cascada)
Población por continente
Países con más o menos de n habitantes
Países con más o menos de n superficie
Calcular población por continente
Mostrar resultados por pantalla
Cancelar y volver al menú
8) Eliminar País
Pedir Nombre y validar datos
Confirmar dos veces
Eliminar de la lista
9) Salir

Organización del Programa

El programa se organiza en un flujo de operaciones que asegura que cada acción esté claramente definida y validada.

El recorrido básico es el siguiente:

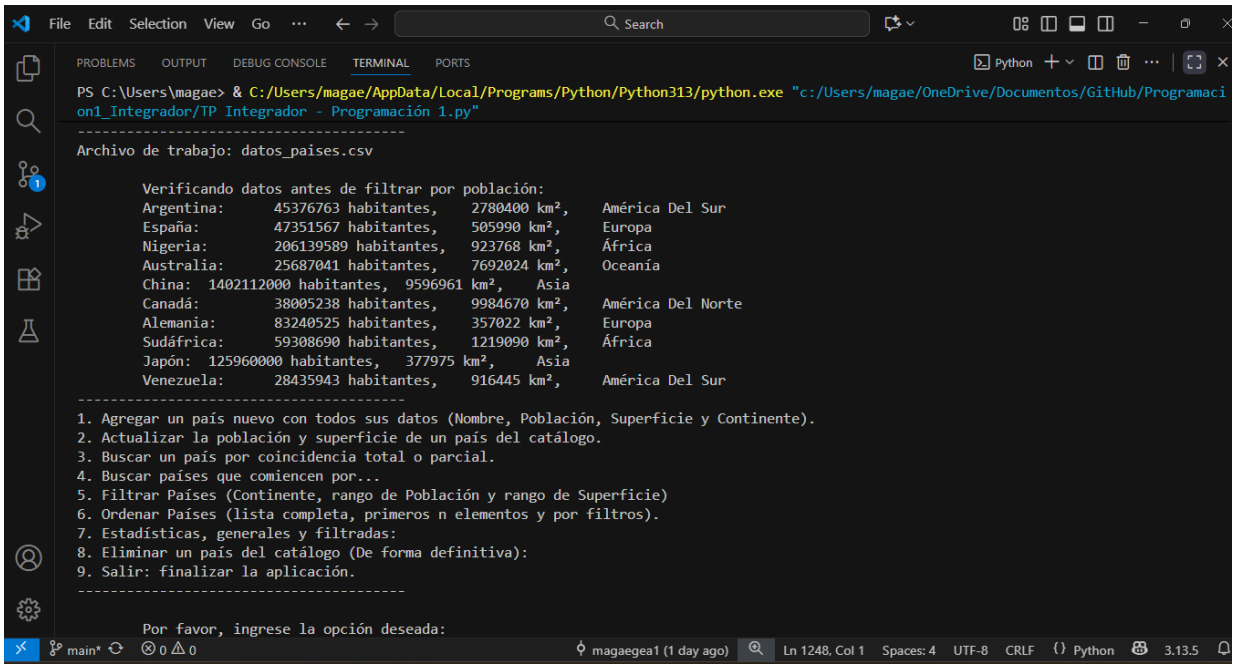
1. Inicio del programa

1. Se muestra el menú principal en consola.
2. Se carga el catálogo desde el archivo CSV, validando estructura y contenido.

2. Selección de opción en el menú

El usuario elige entre las funcionalidades disponibles:

1. Agregar un país
2. Actualizar población y superficie
3. Buscar país por coincidencia parcial o exacta
4. Buscar países que comiencen por…
5. Filtrar países (continente, población, superficie)
6. Ordenar países (nombre, población, superficie, continente)
7. Estadísticas (promedios, máximos, mínimos, cantidad por continente)
8. Eliminar un país (extra que agregaste, muy bien justificado)
9. Salir (finalizar la aplicación)



3. Validación de entradas

Para garantizar la robustez del sistema y evitar errores en el catálogo de países, se implementaron funciones específicas de validación. Estas funciones actúan como filtros previos: no permiten que datos inválidos entren en la lista de diccionarios ni en el archivo CSV.

El módulo csv está diseñado para manejar la lectura y escritura de archivos de manera sencilla y confiable (Python Software Foundation, 2024). En nuestro programa, esta referencia fue clave para implementar la función guardar_catalogo_en_csv() y centralizar la persistencia de datos.

Las validaciones se organizan en tres grandes grupos:

I. Validación de texto

preparar_texto_valido() Asegura que el texto ingresado no esté vacío ni sea numérico. Normaliza el formato con .title() para mantener coherencia en nombres propios.

preparar_texto_normalizado(texto) Elimina acentos y convierte a minúsculas, permitiendo comparaciones insensibles a mayúsculas y caracteres especiales. Se usa para evitar duplicados y facilitar búsquedas.

```
PS C:\Users\magae> & C:/Users/magae/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/ma
on1_Integrador/TP Integrador - Programación 1.py"

        Por favor, ingrese la opción deseada: 1

        1. Agregar un país
        Debe indicar:
        - Nombre del país
        - Población (entero positivo)
        - Superficie en km² (entero positivo)
        - Continente

        Por favor, indique el nombre del país que desea cargar:
        Disculpe, la entrada no puede estar vacía, intente nuevamente: 1234
        Disculpe, no puede ser un número, intente nuevamente:           saN      MARinO

        Por favor, indique la Población de San Marino:
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente: -4
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente: 0
        Disculpe, debe no puede ser 0 (cero). Intente nuevamente: asdf
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente:          35436

        Por favor, indique la Superficie en km² de San Marino:
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente: asdf
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente: -4
        Disculpe, debe ingresar un número entero positivo. Intente nuevamente: 0
        Disculpe, debe no puede ser 0 (cero). Intente nuevamente: 61
```

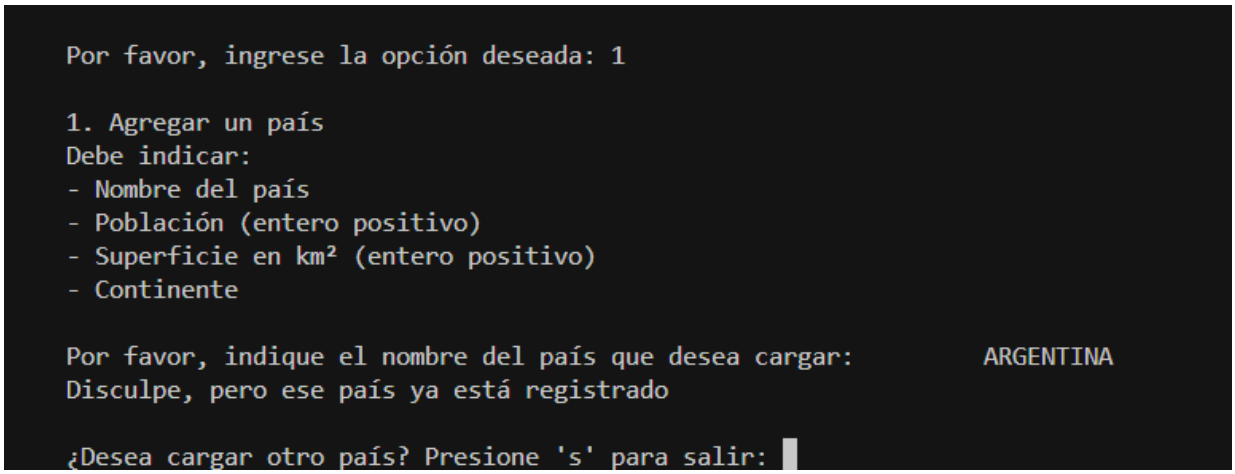
II. Validación de números

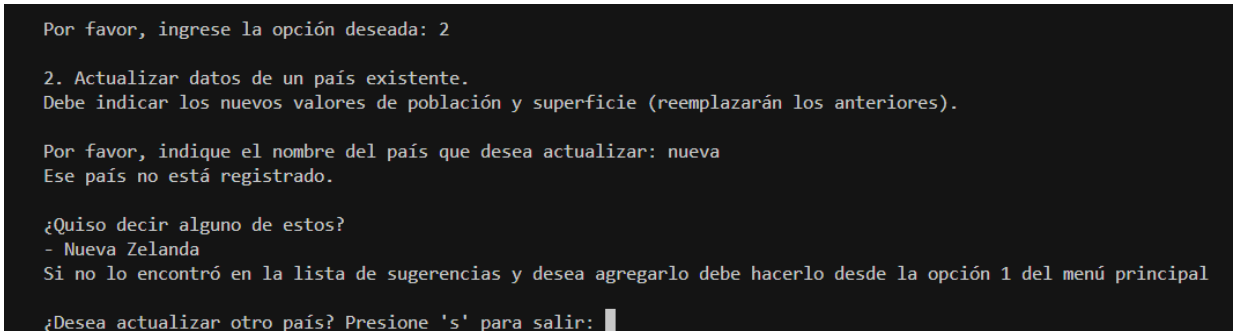
preparar_entero_positivo() Garantiza que la entrada sea un número entero positivo. Rechaza valores vacíos, no numéricos o iguales a cero. Se aplica en campos de población y superficie.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\magae> & C:/Users/magae/AppData/Local/Programs/Python/Python313/python.exe
on1_Integrador/TP Integrador - Programación 1.py"
6. Ordenar Países (lista completa, primeros n elementos y por filtros).
7. Estadísticas, generales y filtradas:
8. Eliminar un país del catálogo (De forma definitiva):
9. Salir: finalizar la aplicación.
----------------------------------------

        Por favor, ingrese la opción deseada:

        Disculpe, pero la opción ingresada no es válida, intente nuevamente...

        Por favor, ingrese la opción deseada: as

        Disculpe, pero la opción ingresada no es válida, intente nuevamente...

        Por favor, ingrese la opción deseada: 0

        Disculpe, pero la opción ingresada no es válida, intente nuevamente...

        Por favor, ingrese la opción deseada: -4

        Disculpe, pero la opción ingresada no es válida, intente nuevamente...

        Por favor, ingrese la opción deseada: 10

        Disculpe, pero la opción ingresada no es válida, intente nuevamente...

        Por favor, ingrese la opción deseada:
```

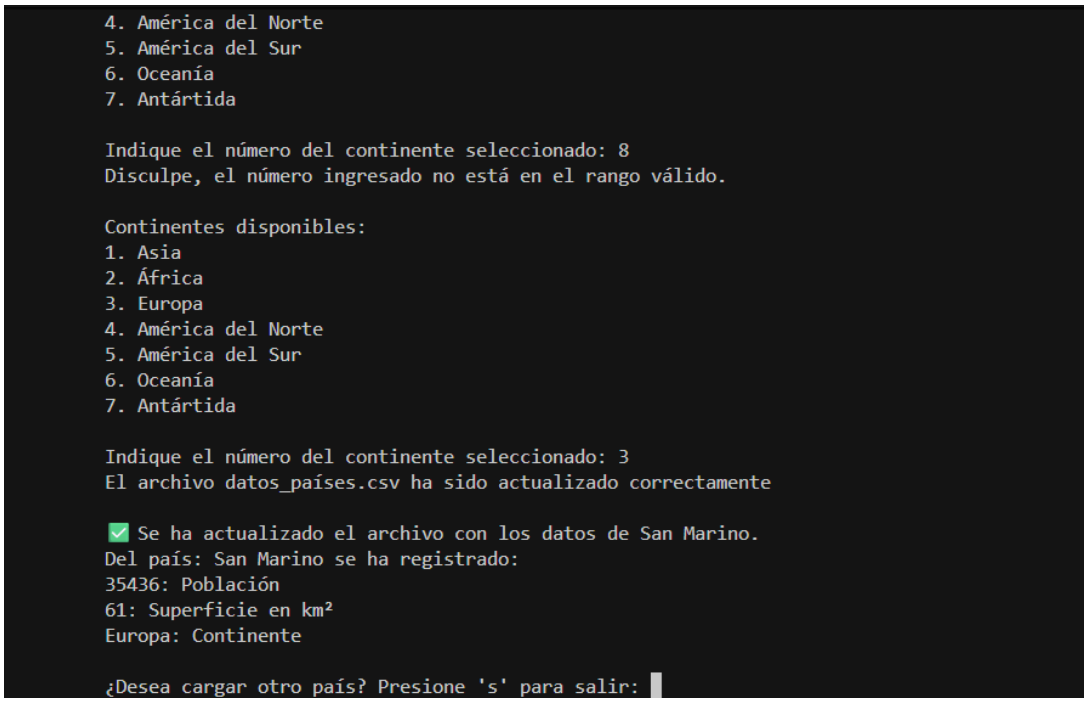
III. Validación de existencia y selección

preparar_pais_ya_existe(lista_paises, nuevo_pais) Comprueba si un país ya está registrado en el catálogo, evitando duplicaciones.





preparar_continentes() Ofrece una lista cerrada de continentes, seleccionados por número. Esto elimina ambigüedades y asegura uniformidad en los registros.



Gracias a estas validaciones, el programa cumple con la consigna de evitar campos vacíos, controlar errores de formato y mostrar mensajes claros de éxito o error.

4. Funcionalidades del Menú

El menú principal organiza todas las acciones del programa en nueve opciones numeradas. Cada opción está implementada como una función independiente, cumpliendo la regla de modularidad (una función = una responsabilidad).

Opción 1 – Agregar país

Función: agregar_pais(lista_paises)

Permite ingresar un nuevo país con nombre, población, superficie y continente.

Valida duplicados y entradas vacías.

Actualiza la lista y el archivo CSV.

```
Por favor, ingrese la opción deseada: 1

1. Agregar un país
Debe indicar:
- Nombre del país
- Población (entero positivo)
- Superficie en km² (entero positivo)
- Continente

Por favor, indique el nombre del país que desea cargar:           NUEVA    ZElanda

Por favor, indique la Población de Nueva Zelanda: 5430000

Por favor, indique la Superficie en km² de Nueva Zelanda: 267700

Por favor, indique el Continente donde se encuentra Nueva Zelanda:
Continentes disponibles:
1. Asia
2. África
3. Europa
4. América del Norte
5. América del Sur
6. Oceanía
7. Antártida

Indique el número del continente seleccionado: 6
El archivo datos_países.csv ha sido actualizado correctamente

✅ Se ha actualizado el archivo con los datos de Nueva Zelanda.
Del país: Nueva Zelanda se ha registrado:
5430000: Población
267700: Superficie en km²
Oceanía: Continente

¿Desea cargar otro país? Presione 's' para salir:
```

Opción 2 – Actualizar país

Función: actualizar_pais(lista_paises)

Modifica población y superficie de un país existente.

Permite actualizar el continente opcionalmente.

Guarda los cambios en el CSV.

```
2. Actualizar datos de un país existente.
Debe indicar los nuevos valores de población y superficie (reemplazarán los anteriores).

Por favor, indique el nombre del país que desea actualizar: nueva zelanda

Datos actuales de Nueva Zelanda:
Población: 5430000
Superficie: 267700 km²
Continente: Oceanía

Por favor, indique la nueva Población de Nueva Zelanda: 543001

Por favor, indique la nueva Superficie en km² de Nueva Zelanda: 267700

¿Desea actualizar también el continente? (s/n): n
El archivo datos_países.csv ha sido actualizado correctamente

✅ Se ha actualizado el archivo con los datos de Nueva Zelanda.
Del país: Nueva Zelanda, tenemos estos datos:
543001: Población
267700: Superficie en km²
Oceanía: Continente

¿Desea actualizar otro país? Presione 's' para salir:
```

```
Por favor, indique el nombre del país que desea actualizar: aeiou

Datos actuales de Aeiou:
Población: 1
Superficie: 1 km²
Continente: Asia

Por favor, indique la nueva Población de Aeiou: 2

Por favor, indique la nueva Superficie en km² de Aeiou: 2

¿Desea actualizar también el continente? (s/n): s

Continentes disponibles:
1. Asia
2. África
3. Europa
4. América del Norte
5. América del Sur
6. Oceanía
7. Antártida

Indique el número del continente seleccionado: 2
Del país: Aeiou, se actualizó este continente: África

El archivo datos_países.csv ha sido actualizado correctamente
```

Opción 3 – Buscar país por nombre

Función: buscar_pais_por_nombre(lista_paises)

Busca coincidencias parciales o exactas en nombres.

Muestra resultados ordenados alfabéticamente.

```
3. Cuál es el país que desea buscar? Puede ser una coincidencia parcial o exacta: an
Hemos encontrado lo siguiente:
2. Alemania:    83240525 habitantes,    357022 Km²,     en Europa
5. Canadá:      38005238 habitantes,    9984670 Km²,    en América Del Norte
7. España:      47351567 habitantes,    505990 Km²,     en Europa
10. Nueva Zelanda:      543001 habitantes,      267700 Km²,     en Oceanía
11. San Marino:         35436 habitantes,       61 Km²,         en Europa

¿Desea hacer otra búsqueda? Presione 's' para salir:
```

Opción 4 – Buscar países que comienzan con…

Función: buscar_paises_que_comienzan_con(lista_paises)

Filtra países cuyo nombre empieza con el texto ingresado.

Útil para búsquedas rápidas por iniciales, es una opción extra a las consignas.

```
4. Ingrese la letra o texto inicial del país que desea buscar: a
Hemos encontrado lo siguiente:
1. Aeiou:       2 habitantes,   2 Km²,  en África
2. Alemania:    83240525 habitantes,    357022 Km²,     en Europa
3. Argentina:   45376763 habitantes,    2780400 Km²,    en América Del Sur
4. Australia:   25687041 habitantes,    7692024 Km²,    en Oceanía

¿Desea hacer otra búsqueda? Presione 's' para salir:
Continuamos...

4. Ingrese la letra o texto inicial del país que desea buscar: c
Hemos encontrado lo siguiente:
5. Canadá:      38005238 habitantes,    9984670 Km²,    en América Del Norte
6. China:       1402112000 habitantes,  9596961 Km²,    en Asia

¿Desea hacer otra búsqueda? Presione 's' para salir:
```

Opción 5 – Filtrar países

Función: filtrar_paises(lista_paises)

Aplica filtros en cascada: continente, población, superficie.

Cada filtro se puede omitir o aplicar según el usuario.

```
5. A continuación se le ofrecerá filtrar por una o varias opciones,
elija la que desee (una a la vez) y vaya omitiendo la que no desee usar...

¿Qué filtro desea aplicar ahora?
1. por Continente
2. por Población
3. por Superficie

Por favor, ingrese la opción deseada: 1

Se va a filtrar por continente... Presione 's' para saltar este filtro:

Por favor, indique el continente que desea filtrar:
Continentes disponibles:
1. Asia
2. África
3. Europa
4. América del Norte
5. América del Sur
6. Oceanía
7. Antártida

Indique el número del continente seleccionado: 1
Se encontraron 2 países en Asia.

¿Desea aplicar otro filtro? Presione 's' para salir:
```

```
¿Desea aplicar otro filtro? Presione 's' para salir:
Continuamos...

5. A continuación se le ofrecerá filtrar por una o varias opciones,
elija la que desee (una a la vez) y vaya omitiendo la que no desee usar...

¿Qué filtro desea aplicar ahora?
1. por Continente
2. por Población
3. por Superficie

Por favor, ingrese la opción deseada: 3

¿Desea filtrar por superficie? Presione 's' para saltar este filtro:
Por favor, indique qué tipo de filtro desea aplicar:

1. Países con más de x km²
2. Países con menos de x km²
3. Países con x km²

Por favor, ingrese la opción deseada: 1

Indique el valor límite de km² que desea usar: 500000
Se encontraron 1 países con una superficie mayor que 500000.

¿Desea aplicar otro filtro? Presione 's' para salir:
```

```
¿Desea aplicar otro filtro? Presione 's' para salir: s
Volvemos al menú principal

Resultados filtrados:
1. China:       1402112000 habitantes,  9596961 Km²,    en Asia
```

```
3. Países con x habitantes

Por favor, ingrese la opción deseada: 2

Indique el valor límite de habitantes que desea usar: 700000
Se encontraron 3 países con una población menor que 700000.

¿Desea aplicar otro filtro? Presione 's' para salir: s
Volvemos al menú principal

Resultados filtrados:
1. Aeiou:       2 habitantes,   2 Km²,  en África
2. Nueva Zelanda:       543001 habitantes,      267700 Km²,     en Oceanía
3. San Marino:  35436 habitantes,       61 Km²,         en Europa
```

Opción 6 – Ordenar países

Funciones: ordenar_por_pais, ordenar_por_poblacion, ordenar_por_superficie

Ordena la lista según el criterio elegido.

Permite mostrar todos los países o sólo los primeros n.

```
Por favor, ingrese la opción deseada: 6

6. Por favor, indique bajo qué criterio quiere ordenar la lista completa:
1. País:
2. Población:
3. Superficie:
4. Continente:

Por favor, ingrese la opción deseada: 4
¿Cuántos elementos desea ver?
        1. Todos
        2. Los primeros n elementos
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 1

Lista ordenada:
1. En América Del Norte:        Canadá:         38005238 habitantes,    9984670 Km²
2. En América Del Sur:  Argentina:      45376763 habitantes,    2780400 Km²
3. En América Del Sur:  Venezuela:      28435943 habitantes,    916445 Km²
4. En Asia:     China:  1402112000 habitantes,  9596961 Km²
5. En Asia:     Japón:  125960000 habitantes,   377975 Km²
6. En Europa:   España:         47351567 habitantes,    505990 Km²
7. En Europa:   Alemania:       83240525 habitantes,    357022 Km²
8. En Europa:   San Marino:     35436 habitantes,       61 Km²
9. En Oceanía:  Australia:      25687041 habitantes,    7692024 Km²
10. En Oceanía:         Nueva Zelanda:  543001 habitantes,      267700 Km²
11. En África:  Nigeria:        206139589 habitantes,   923768 Km²
12. En África:  Sudáfrica:      59308690 habitantes,    1219090 Km²
13. En África:  Aeiou:  2 habitantes,   2 Km²
---------------------------------
```

```
Por favor, ingrese la opción deseada: 6

6. Por favor, indique bajo qué criterio quiere ordenar la lista completa:
1. País:
2. Población:
3. Superficie:
4. Continente:

Por favor, ingrese la opción deseada: 1
¿Cuántos elementos desea ver?
        1. Todos
        2. Los primeros n elementos
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 2
¿Cuántos elementos desea ver?: 5
        1. De menor a mayor
        2. De mayor a menor
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 1

Lista ordenada:
1. Aeiou:       2 habitantes,   2 Km²,  en África
2. Alemania:    83240525 habitantes,    357022 Km²,     en Europa
3. Argentina:   45376763 habitantes,    2780400 Km²,    en América Del Sur
4. Australia:   25687041 habitantes,    7692024 Km²,    en Oceanía
5. Canadá:      38005238 habitantes,    9984670 Km²,    en América Del Norte
```

```
6. Por favor, indique bajo qué criterio quiere ordenar la lista completa:
1. País:
2. Población:
3. Superficie:
4. Continente:

Por favor, ingrese la opción deseada: 3
¿Cuántos elementos desea ver?
        1. Todos
        2. Los primeros n elementos
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 2
¿Cuántos elementos desea ver?: 5
        1. De menor a mayor
        2. De mayor a menor
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 2

Lista ordenada:
1. Canadá:      38005238 habitantes,    9984670 Km²,    en América Del Norte
2. China:       1402112000 habitantes,  9596961 Km²,    en Asia
3. Australia:   25687041 habitantes,    7692024 Km²,    en Oceanía
4. Argentina:   45376763 habitantes,    2780400 Km²,    en América Del Sur
5. Sudáfrica:   59308690 habitantes,    1219090 Km²,    en África
```

```
6. Por favor, indique bajo qué criterio quiere ordenar la lista completa:
1. País:
2. Población:
3. Superficie:
4. Continente:

Por favor, ingrese la opción deseada: 4
¿Cuántos elementos desea ver?
        1. Todos
        2. Los primeros n elementos
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 2
¿Cuántos elementos desea ver?: 5
        1. De menor a mayor
        2. De mayor a menor
Seleccione 1 o 2:
Por favor, ingrese la opción deseada: 2

Lista ordenada:
1. Australia:   25687041 habitantes,    7692024 Km²,    en Oceanía
2. Nueva Zelanda:       543001 habitantes,      267700 Km²,     en Oceanía
3. España:      47351567 habitantes,    505990 Km²,     en Europa
4. Alemania:    83240525 habitantes,    357022 Km²,     en Europa
5. San Marino:  35436 habitantes,       61 Km²,         en Europa
```

Opción 7 – Estadísticas

Estas operaciones se apoyan en funciones integradas de Python como max(), min(), sum() y len() (W3Schools, 2024).

Función: mostrar_estadisticas(lista_paises)

Calcula y muestra:

País con mayor y menor población.

Promedio de población.

Promedio de superficie.

Cantidad de países por continente.

```
7. ¿Qué estadística desea calcular?

1. Promedio de población
2. Promedio de superficie
3. País con mayor población
4. País con menor población
5. País con mayor superficie
6. País con menor superficie
7. Volver al menú principal

Por favor, ingrese la opción deseada: 1

¿Desea aplicar filtros antes de calcular el promedio de población?
1. Filtrar por continente
2. Filtrar por rango de población
3. Filtrar por rango de superficie
4. No aplicar filtros (usar lista completa)

Por favor, ingrese la opción deseada: 1

Se va a filtrar por continente... Presione 's' para saltar este filtro:

Por favor, indique el continente que desea filtrar:
Continentes disponibles:
1. Asia
2. África
3. Europa
4. América del Norte
5. América del Sur
6. Oceanía
7. Antártida

Indique el número del continente seleccionado: 1
Se encontraron 2 países en Asia.

✅ Promedio de población calculado sobre 2 países: 764036000.00
```

```
7. ¿Qué estadística desea calcular?

1. Promedio de población
2. Promedio de superficie
3. País con mayor población
4. País con menor población
5. País con mayor superficie
6. País con menor superficie
7. Volver al menú principal

Por favor, ingrese la opción deseada: 5

¿Desea aplicar filtros antes de calcular el máximo de superficie en km²?
1. Filtrar por continente
2. Filtrar por rango de población
3. Filtrar por rango de superficie
4. No aplicar filtros (usar lista completa)

Por favor, ingrese la opción deseada: 2

¿Desea filtrar por población? Presione 's' para saltar este filtro:
Por favor, indique qué tipo de filtro desea aplicar:

1. Países con más de x habitantes
2. Países con menos de x habitantes
3. Países con x habitantes

Por favor, ingrese la opción deseada: 2

Indique el valor límite de habitantes que desea usar: 10000000
Se encontraron 3 países con una población menor que 10000000.

✅ País con mayor superficie en km²: Nueva Zelanda (267700)
```

Opción 8 – Eliminar país

Función: eliminar_pais(lista_paises)

Borra un país del catálogo.

Útil para corregir errores o eliminar duplicados. Esta opción es extra a las solicitada en las consignas.

```
Por favor, ingrese la opción deseada: 8

8. Eliminar un país del catálogo.
¡CUIDADO! Luego de borrar un país del catálogo no podrá deshacer esta acción. Se le pedirá confirmación...
Por favor, indique el nombre del país que desea eliminar: aeiou

Se encontró el país: Aeiou
Población: 2
Superficie: 2 km²
Continente: África

¿Está seguro de que desea eliminar este país? ⚠️ Esta acción no se puede deshacer. (s/n): s

✅ El país Aeiou ha sido eliminado del catálogo.
El archivo datos_países.csv ha sido actualizado correctamente

¿Desea eliminar otro país? Presione 's' para salir: s
Volvemos al menú principal
--------------------------------
```

Opción 9 – Salir

Finaliza la ejecución del programa.

```
----------------------------------------
1. Agregar un país nuevo con todos sus datos (Nombre, Población, Superficie y Continente).
2. Actualizar la población y superficie de un país del catálogo.
3. Buscar un país por coincidencia total o parcial.
4. Buscar países que comiencen por...
5. Filtrar Países (Continente, rango de Población y rango de Superficie)
6. Ordenar Países (lista completa, primeros n elementos y por filtros).
7. Estadísticas, generales y filtradas:
8. Eliminar un país del catálogo (De forma definitiva):
9. Salir: finalizar la aplicación.
----------------------------------------

        Por favor, ingrese la opción deseada: 9
¡Gracias por usar el sistema!
¡Hasta Pronto!
PS C:\Users\magae>
```

Cada opción del menú está respaldada por validaciones y mensajes claros, lo que asegura que el usuario pueda interactuar de manera sencilla y confiable.

Validaciones y manejo de errores

Validaciones implementadas

Texto válido:

- preparar_texto_valido() asegura que el usuario no ingrese cadenas vacías ni números como nombre de país.
- Normaliza con .title() y elimina espacios redundantes.

Enteros positivos:

- preparar_entero_positivo() valida que población y superficie sean números enteros positivos, distintos de cero.

Duplicados:

- preparar_pais_ya_existe() compara nombres normalizados para evitar cargar dos veces el mismo país.

Continentes válidos:

- preparar_continentes() ofrece una lista cerrada de opciones, evitando ambigüedades.

Opciones de menú:

- pedir_opcion_1_a_n(n) garantiza que el usuario solo pueda elegir entre valores válidos.

Normalización de texto:

- preparar_texto_normalizado() elimina acentos y caracteres especiales para búsquedas y comparaciones consistentes.

Manejo de errores

- CSV: cargar_datos_desde_csv() controla si el archivo existe; si no, lo crea con encabezado.

- Omite registros inválidos (población/superficie no numéricas o ≤0) y muestra un mensaje de advertencia.

Entradas inválidas:

- Mensajes claros al usuario cuando se ingresa un valor incorrecto (ej. “Debe ingresar un número entero positivo”).

Búsquedas sin resultados:

- Mensajes como “Disculpe, no se encontró ninguna coincidencia” y sugerencias de países similares.

Filtros sin coincidencias:

- Mensajes como “⚠️ No hay países que cumplan con los filtros seleccionados”.

Eliminación con confirmación:

- eliminar_pais() pide confirmación explícita antes de borrar un país, evitando errores irreversibles.

Resultados Obtenidos.

1. ¿Qué funcionó bien?

Funcionó muy bien la estructura modular del programa, que permitió desarrollar cada función como una unidad autónoma y clara. La lectura del archivo CSV fue estable, y logramos mostrar, filtrar, ordenar y analizar los datos sin errores. El menú principal se convirtió en una compuerta ética que guía al usuario con fluidez, y cada función fue documentada con docstrings que integran precisión técnica y sensibilidad simbólica. La validación de entradas y la separación de responsabilidades hicieron que el código sea legible, mantenible y coherente con la consigna.

2. ¿Qué cosas nos causaron problemas?

No hubo dificultades con la carga ni visualización de datos. Desde el inicio, organizar el código en funciones bien delimitadas facilitó la prueba y el ajuste. Las estructuras de datos (listas de diccionarios) respondieron como se esperaba, y las validaciones básicas evitaron errores comunes. La decisión de centralizar la escritura al CSV en una única función (guardar_catalogo_en_csv) fue clave para mantener el control y cumplir con la consigna.

3. ¿Errores o decisiones que cambiaron durante el desarrollo?

Al principio hubo dudas sobre cómo estructurar los diccionarios y cómo leer el CSV sin romper la lógica del programa. También fue necesario eliminar funciones que escribían directamente al archivo, y reemplazarlas por una estrategia más ética y controlada. Se ajustaron nombres de funciones, se reescribieron validaciones y se incorporaron bifurcaciones simbólicas en el menú para mejorar la experiencia del usuario, también se cambió el filtro a un modelo de cascada y se decidió agregar la función de eliminar para poder corregir errores que se generan en el archivo CSV cuando hay, por ejemplo, errores de tipeo. Cada error fue resignificado como oportunidad de aprendizaje y reparación.

4. ¿Cuál fue el mayor desafío? ¿Cómo fue la comunicación del equipo?

El mayor desafío fue lograr tener todas las funciones necesarias de las validaciones claras, así como funciones auxiliares que se fueron agregando o sacando. Además, lograr que todas las funciones se integren sin romper la coherencia del flujo: que cada bifurcación del menú condujera a una acción clara, sin repeticiones ni ambigüedades. También fue un reto mantener la documentación viva y alineada con cada ajuste técnico. En cuanto al trabajo en equipo, la comunicación fue respetuosa y complementaria: ambas trabajamos en el desarrollo del código, las validaciones, las pruebas y la documentación técnica/simbólica, de la parte teórica, el PDF y el video. Compartimos avances, revisamos juntas y celebramos cada brote de autonomía. También utilizamos la IA: Copilot, para pensar algunas ideas y revisar algunas partes del TP.

Conclusiones finales

¿Qué aprendimos?

Durante este trabajo profundizamos nuestros conocimientos al aplicar todo lo que vimos en la materia de una forma práctica y real. Consolidamos mejor cómo usar listas, diccionarios, funciones y estructuras de control para crear un programa completo. También valoramos lo importante que es tener un código ordenado, dividido por partes y fácil de entender. Al mismo tiempo, al realizar investigación teórica, aprendimos a buscar información confiable específica, leer documentación oficial y mejoramos nuestra capacidad de síntesis, lo que nos ayudó a comprender más a fondo cómo funciona Python.

¿Por qué nos parecen importantes estas herramientas?

Nos parecen herramientas clave porque son la base de cualquier proyecto de programación. Sirven para organizar datos, automatizar tareas y analizar información de una forma más clara y eficiente. Además, son conocimientos que se aplican en el mundo real, no solo en la materia. Aprender a usarlas bien nos da una base sólida para seguir avanzando en otros lenguajes o proyectos más grandes.

Justificación del uso de las herramientas (con ejemplos):

Usamos listas para almacenar varios países, diccionarios para guardar los datos de cada uno (nombre, población, superficie y continente), y funciones para dividir las distintas partes del programa, como buscar, filtrar o calcular estadísticas. También trabajamos con archivos CSV, que nos permitieron guardar y leer la información fácilmente, sin tener que escribir todo cada vez.

¿Cómo organizamos el trabajo entre las dos?

Nos dividimos el trabajo según nuestras fortalezas. Magaly se encargó del código, las validaciones y las pruebas del sistema, mientras que yo hice la parte teórica, el PDF, el video y el README. Además, buscamos información y ejemplos por fuera para complementar lo aprendido en clase. Nos mantuvimos en contacto todo el tiempo, revisando juntas cada parte y ayudándonos cuando algo no funcionaba. Esa comunicación y colaboración constante fue lo que hizo que el proyecto salga bien.

En resumen, este trabajo nos ayudó a unir teoría y práctica, a reforzar conocimientos de programación y a desarrollar habilidades blandas como la organización, la comunicación y la búsqueda autónoma de información.

Referencias bibliográficas (formato APA 7)

Oracle. (2024). What is data management? Oracle Corporation. https://www.oracle.com/database/what-is-data-management/

Python Software Foundation. (2024). csv — CSV file reading and writing. En Python Documentation. https://docs.python.org/3/library/csv.html

Python Software Foundation. (2024). Sorting HOW TO. En Python Documentation. https://docs.python.org/3/howto/sorting.html

Python Software Foundation. (2024). The for statement. En Python Documentation. https://docs.python.org/3/tutorial/controlflow.html#for-statements

Python Software Foundation. (2024). The if statement. En Python Documentation. https://docs.python.org/3/tutorial/controlflow.html#if-statements

SAP SE. (2024). What is data management? https://www.sap.com/slovenia/products/technology-platform/what-is-data-management.html

Sweigart, A. (2019). Automate the boring stuff with Python (2nd ed., Capítulos 4–5). No Starch Press. https://automatetheboringstuff.com/

W3Schools. (2024). Python dictionaries. https://www.w3schools.com/python/python_dictionaries.asp

W3Schools. (2024). Python dictionary – glossary. https://www.w3schools.com/python/gloss_python_dictionary.asp

W3Schools. (2024). Python file handling / CSV files. https://www.w3schools.com/python/python_file_handling.asp

W3Schools. (2024). Python for loops & while loops. https://www.w3schools.com/python/python_for_loops.asp

W3Schools. (2024). Python functions. https://www.w3schools.com/python/python_functions.asp

W3Schools. (2024). Python if...else statements.
https://www.w3schools.com/python/python_conditions.asp

W3Schools. (2024). Python lists. https://www.w3schools.com/python/python_lists.asp

W3Schools. (2024). Python sort lists.
https://www.w3schools.com/python/python_lists_sort.asp