



**TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA**

PROGRAMACIÓN 1
Trabajo Práctico Integrador

Alumnos

Emiliano Veneroso - C12

Mario Martini - C 8

Docentes Titulares

Ariel Enferrel - C12

Sebastián Bruselario - C8

Docente Tutores

Franco Gonzalez - C12

Virginia Cimino - C8

noviembre 2025

Tabla de contenido

Consignas	3
Requerimientos técnicos	
Informe teórico de conceptos aplicados.	5
Esquema de funcionalidades.	6
Conclusiones	9
Referencias	10

Introducción

En el presente trabajo se desarrolla una aplicación en lenguaje Python orientada a la gestión de información sobre países, con el propósito de poner en práctica los principales conceptos aprendidos en la asignatura Programación 1.

El proyecto busca integrar el uso de estructuras de datos como listas y diccionarios, junto con la modularización del código mediante funciones, la implementación de estructuras condicionales y repetitivas, y la aplicación de técnicas de filtrado, ordenamiento y cálculo de estadísticas básicas.

El sistema diseñado permite leer y almacenar información desde un archivo CSV, ofrecer consultas dinámicas mediante menús, y generar indicadores sobre población, superficie y distribución geográfica por continente.

A través de esta práctica se refuerza la capacidad para organizar datos, aplicar lógica de control, y estructurar soluciones reutilizables y escalables, fomentando un estilo de programación claro, ordenado y orientado a la resolución de problemas concretos.

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.

El objetivo principal es afianzar el uso de estructuras de datos, modularización con funciones y técnicas de filtrado/ordenamiento, aplicando los conceptos aprendidos en Programación 1.

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:

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)

- Explicar en un informe teórico los conceptos aplicados:
 - Listas

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

- Definir el flujo de operaciones principales en un diagrama o esquema.

2. Funcionalidades mínimas del sistema

El programa debe ofrecer un menú de opciones en consola que permita:

- Agregar un país con todos los datos necesarios para almacenarse (No se permiten campos vacíos).
- Actualizar los datos de Población y Superficie de un País.
- Buscar un país por nombre (coincidencia parcial o exacta).
- Filtrar países por:
 - Continente
 - Rango de población
 - Rango de superficie
- Ordenar países por:
 - Nombre
 - Población
 - Superficie (ascendente o descendente)
- Mostrar estadísticas:
 - País con mayor y menor población
 - Promedio de población
 - Promedio de superficie
 - Cantidad de países por continente

3. Validaciones

- Controlar errores de formato en el CSV.
- Evitar fallos al ingresar filtros inválidos o búsquedas sin resultados.

- Mensajes claros de éxito/error.

Informe teórico de conceptos aplicados.

LISTAS: En este trabajo se utiliza una lista principal llamada lista_paises, que contiene un conjunto de diccionarios, donde cada diccionario representa un país.

Esto permite recorrer fácilmente los elementos, aplicar filtros, realizar ordenamientos y obtener estadísticas mediante iteraciones y condiciones.

DICCIONARIOS: En el sistema, cada país se modela como un diccionario con las claves: "nombre", "poblacion", "superficie" y "continente".

Esta estructura es ideal para representar entidades con atributos, ya que mejora la claridad y evita depender de posiciones fijas como en una lista común.

FUNCIONES: El código del proyecto está fuertemente modularizado: existen funciones para validar datos (es_entero_positivo), para mostrar información (mostrar_paises), para filtrar (filtrar_por_continente), ordenar (ordenar_paises) y generar estadísticas (mostrar_promedio_poblacion, entre otras).

Esto mejora la legibilidad, el mantenimiento y favorece la reutilización del código.

CONDICIONALES: Las estructuras condicionales (if, elif, else) se usan para tomar decisiones dentro del flujo del programa.

Se aplican, por ejemplo, al validar si un país ya existe antes de agregarlo, al controlar entradas incorrectas o al determinar qué opción ejecutar según la selección del usuario en el menú principal.

ORDENAMIENTOS: El ordenamiento organiza los datos según un criterio determinado.

En este trabajo se implementa un ordenamiento por inserción (Insertion Sort) y también versiones basadas en sorted(), para ordenar los países por nombre, población o superficie, tanto en forma ascendente como descendente.

Esto permite presentar los resultados de manera clara y coherente para el usuario.

ESTADISTICAS BASICAS: Las estadísticas permiten obtener información general sobre los datos cargados.

El programa calcula:

El país con mayor y menor población,

El promedio de población y superficie,

Y la cantidad de países por continente.

Estas operaciones usan sumatorias, contadores y funciones auxiliares (max, min, sum, len o bucles for) para recorrer la lista de países y generar indicadores significativos.

ARCHIVO CSV: El formato CSV (Comma-Separated Values) se utiliza para almacenar los datos de forma persistente y legible.

El programa implementa lectura y escritura mediante el módulo estándar csv de Python, permitiendo guardar los países agregados o modificados sin perder información entre ejecuciones.

La función cargar_datos_csv() abre el archivo, valida los campos y convierte los valores numéricos; mientras que guardar_datos_csv() sobrescribe el archivo con los datos actualizados.

Esto garantiza una persistencia sencilla y portable, compatible con planillas de cálculo o bases de datos futuras.

Esquema de funcionalidades.

Estructura del Código: Gestión de Países

El código implementa un sistema para Leer, Crear y Actualizar datos desde un archivo CSV ; junto con funcionalidades de Búsqueda, Filtrado, Ordenamiento y Estadísticas. Utilizando la sobreecritura del archivo CSV (paises.csv) para la persistencia de datos.

Módulos del Programa:

I. Importación y Configuración Inicial

import csv: Para la lectura y escritura de archivos CSV.

import os: Para verificar la existencia del archivo (os.path.exists).

nombre_archivo = "paises.csv": Define el nombre del archivo de datos.

II. Módulo de Validaciones

Conjunto de funciones para asegurar valores numéricos (poblacion, superficie).

es_entero_positivo(valor_str):

validar_entero(dato):

validar_entero_desde_archivo(valor_str):

III. Módulo de Visualización y Persistencia de datos (CSV)

mostrar_paises(paises):

cargar_datos_csv(nombre_archivo): Guarda la lista completa de países en el archivo CSV, sobrescribiendo el contenido.

IV. Módulo de Gestión

agregar_pais(lista_paises, nombre_archivo): Solicita datos, valida el nombre (que no esté vacío y no exista), valida población y superficie con validar_entero. Agrega el nuevo país a lista_paises y llama a guardar_datos_csv.

actualizar_datos(lista_paises, nombre_archivo): Pide el nombre de un país. Si lo encuentra, permite modificar la población y la superficie (opcionalmente) utilizando validar_entero_desde_archivo para la validación de las nuevas entradas. Llama a guardar_datos_csv si se realizó algún cambio válido.

buscar_pais(lista_paises): Busca países por nombre (coincidencia parcial e insensible a mayúsculas/minúsculas). Muestra los resultados encontrados con mostrar_paises.

V. Módulo de Ordenamiento y Filtrado

1. Ordenamiento

ordenar_lista(lista):

opción_ordenamiento():

ordenar_paises(lista_paises, clave, opcion_orden): Ordena y muestra la lista de países según una clave ('nombre', 'poblacion', 'superficie') y una opción de orden (ascendente/descendente). Llama a mostrar_paises.

menu_ordenamiento(lista_paises): Submenú que permite al usuario elegir el criterio de ordenamiento y el sentido.

2. Filtrado

filtrar_por_continente(lista_paises): Obtiene y muestra los continentes únicos disponibles. Filtra los países que coinciden con el continente buscado. Llama a mostrar_paises.

filtrar_por_rango_poblacion(lista_paises): Solicita y valida el rango (mínimo y máximo) de población. Filtra los países en ese rango. Llama a mostrar_paises.

filtrar_por_rango_superficie(lista_paises): Solicita y valida el rango de superficie. Filtra los países. Llama a mostrar_paises.

menu_filtros(lista_paises): Submenú que permite al usuario elegir el tipo de filtro a aplicar.

VI. Módulo de Estadísticas

obtener_extremos_poblacion(lista_paises): Encuentra y muestra el país con la mayor y menor población.

calcular_promedio(lista_paises, clave): Función auxiliar que calcula el promedio de una clave numérica.

mostrar_promedio_poblacion(lista_paises): Muestra el promedio de población (usa calcular_promedio).

mostrar_promedio_superficie(lista_paises): Muestra el promedio de superficie (usa calcular_promedio).

contar_por_continente(lista_paises): Cuenta la cantidad de países en cada continente y muestra el resumen.

menu_estadisticas(lista_paises): Submenú que permite al usuario elegir la estadística a visualizar.

VII. Flujo Principal del Programa

Carga Inicial: Se ejecuta lista_paises = cargar_datos_csv(nombre_archivo) para cargar o inicializar la lista de países desde el CSV.

Menú Principal (Bucle while True):

Muestra las opciones principales del sistema,

1. Ingresar títulos (múltiples).

2. Ingresar ejemplares.

3. Mostrar catálogo.

4. Consultar disponibilidad.

5. Listar agotados.

6. Agregar título.

7. Actualizar ejemplares (préstamo/devolución).

8. Salir.

Pide una opción al usuario,

Utiliza una sentencia `match` para invocar la función correspondiente, pasando `lista_paises` y `nombre_archivo` (si es necesario) como argumentos.

Opción 8 (Salir): Llama a `guardar_datos_csv` y sale del bucle con `break`.

Conclusiones

El desarrollo de este trabajo permitió afianzar los conocimientos fundamentales de programación estructurada en Python, aplicando de forma integrada los conceptos de listas, diccionarios, funciones, validaciones y manejo de archivos CSV.

La elaboración de un sistema funcional, con un menú interactivo y múltiples operaciones sobre datos reales, facilitó la comprensión del flujo lógico de un programa y de la importancia de la modularización para mantener un código legible, flexible y fácil de mantener.

Además, se evidenció el valor de las estructuras de control y los procesos de filtrado y ordenamiento para analizar información y generar estadísticas significativas.

Referencias

Universidad Tecnológica Nacional (1ra de.). (2025). *Tecnicatura Universitaria en Programación a Distancia. Programación 1 (Unidades 1 a 8)* . Universidad Tecnológica Nacional.

Python Software Foundation. (s. f.). *csv — Lectura y escritura de ficheros CSV*.

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

Python Software Foundation. (s. f.). *os — Interfaces misceláneas del sistema operativo*.

Recuperado de <https://docs.python.org/es/3/library/os.html#module-os>

Python Software Foundation. (s. f.). *Búsqueda en la Documentación de Python*.

Recuperado de <https://docs.python.org/es/3/search.html?q=enumerate>

Link al video de la presentación : <https://www.youtube.com/watch?v=qdSet6bM9mE>

Link al repositorio GitHub : https://github.com/Mario-Martini-25/TPI_P1