T.P. Integrador – Programación 1

Tema: Gestión de Datos de Países en Python

Carrera: Tecnicatura Universitaria en Programación

Integrantes:

Adrián González, Comisión 6

Ezequiel Taboada, Comisión 13

Fecha: septiembre 2025

Introducción

El presente trabajo práctico integrador tiene como objetivo aplicar los conceptos fundamentales de la materia Programación 1 mediante el desarrollo de una aplicación en Python que permite gestionar información sobre países. El sistema implementa operaciones de búsqueda, filtrado, ordenamiento y generación de estadísticas a partir de un archivo CSV, integrando el uso de listas, diccionarios, funciones, estructuras de control y modularización del código.
En esta nueva versión se incorporó una interfaz gráfica con Tkinter para la carga de archivos, lo cual mejora la experiencia del usuario. También se fortaleció el control de errores y se amplió la organización modular del sistema.

Objetivos

- Implementar un sistema de gestión de países con Python.
- Reforzar el uso de estructuras de datos como listas y diccionarios.
- Practicar la modularización del código dividiendo el sistema en distintos archivos.
- Aplicar manejo de errores y validaciones en la carga y procesamiento de datos.
- Incorporar una interfaz gráfica para la selección de archivos.
- Desarrollar estadísticas y operaciones útiles sobre la información.

Descripción del sistema

El sistema permite al usuario gestionar información de países contenida en un archivo CSV. Ofrece las siguientes funcionalidades:

1. Cargar archivo CSV mediante un cuadro de diálogo gráfico.
2. Agregar un país.
3. Editar población y/o superficie.
4. Buscar países por coincidencia parcial o exacta en el nombre.
5. Filtrar países por continente, rango de población o superficie.
6. Ordenar países por nombre, población o superficie, en orden ascendente o descendente.
7. Mostrar estadísticas: mayor y menor población, mayor y menor superficie, promedios de población y superficie por continente, y cantidad de países por continente.
8. Eliminar un país.
9. Salir del programa.

Todas estas operaciones se realizan de manera interactiva mediante menús en consola.

Diseño del programa

El sistema se diseñó con una estructura modular, dividiéndose en distintos archivos que cumplen responsabilidades específicas:

- a_main.py: control principal del flujo y menús.
- b_funciones_csv.py: funciones de carga de archivos CSV.
- c_funciones_colecciones.py: operaciones sobre colecciones de países (filtros, ordenamientos, etc.).
- d_modelo.py: definición del modelo de datos de un país.
- e_funciones_generales.py: utilidades generales para interacción con el usuario y visualización.
- paises.csv: dataset inicial con la información base.

Este diseño modular mejora la organización, legibilidad y mantenibilidad del código.

Flujo de operaciones principales

El flujo de operaciones del sistema sigue los siguientes pasos:

1. Inicio del programa.
2. Selección del archivo CSV mediante interfaz gráfica.
3. Carga de datos en memoria.
4. Presentación del menú principal con opciones de agregar, Actualizar, búsqueda, filtrado, ordenamiento, estadísticas, eliminar y salida.
5. Ejecución de la opción seleccionada.
6. Retorno al menú principal hasta que el usuario seleccione 'Salir'.
7. Fin del programa.

Diagrama de flujo simplificado:
Inicio → Cargar CSV → Menú Principal → [Agregar | Actualizar | Buscar | Filtrar | Ordenar | Estadísticas | Mostrar | Eliminar] → ¿Salir? → Sí → Fin / No → Volver al menú.

Implementación

El sistema fue implementado en Python 3.x utilizando exclusivamente librerías estándar. La interfaz gráfica para la carga de archivos se resolvió con Tkinter. Cada módulo contiene funciones específicas, aplicando principios de responsabilidad única. Se realizaron validaciones de entradas y un control robusto de posibles fallas para garantizar la fiabilidad del programa.

Conclusiones

Este trabajo nos permitió descubrir y aplicar herramientas fundamentales del lenguaje Python. Aprendimos a utilizar listas y diccionarios para organizar datos, funciones para dividir el código en partes reutilizables, y estructuras condicionales para tomar decisiones dentro del programa. También exploramos cómo realizar ordenamientos, aplicar filtros y generar estadísticas básicas, lo que nos ayudó a entender cómo se puede analizar información de manera automatizada.

Estas herramientas nos parecen importantes porque permiten resolver problemas reales de forma eficiente. Por ejemplo, poder filtrar países por continente o calcular el promedio de población nos mostró cómo la programación puede facilitar tareas que serían muy tediosas de hacer manualmente.

Justificamos el uso de estas herramientas porque cada una cumple un rol específico en el sistema: las listas almacenan los países, los diccionarios representan sus atributos, las funciones organizan el código, y las estructuras condicionales permiten controlar el flujo del programa. Un ejemplo claro fue el uso de sorted() con claves personalizadas para ordenar países por superficie o población, lo que nos permitió obtener resultados precisos con pocas líneas de código.

La organización del trabajo entre nosotros fue clave. Dividimos las tareas de forma equitativa: uno se encargó del diseño del sistema y la estructura del código, mientras que el otro se ocupó de la carga de datos, pruebas y documentación. Esta división nos permitió avanzar de manera ordenada y aprender colaborativamente, compartiendo dudas y soluciones a lo largo del desarrollo.

En resumen, este proyecto nos ayudó a comprender cómo aplicar la programación a un caso práctico, y nos motivó a seguir explorando.

Bibliografía

- **Principiantes de Python**. *Archivos CSV — Lectura y escritura de archivos CSV*.
- **Keepcoding.** *Tkinter — Interfaz gráfica de usuario*.
- **Recursos Python**. *Examinar archivo o carpeta en Tkinter*
- **Tutorial oficial de Python en español.** Información oficial de Python.