

Trabajo práctico integrador – Programación I

Sistema de gestión de países

Cortés Oriana Ailen

Profesor: Cintia Rigoni

Tutor: Oscar Londero

Gaston Crespo (Comisión 14)

Profesor: Ariel Enferrel

Tutor: Maximiliano Sar Fernandez

ÍNDICE

1. Introducción
2. Objetivos
3. Marco teórico
4. Diseño previo al código
 - 4.1. Diagrama o esquema del flujo del programa
5. Desarrollo del caso práctico
6. Resultados y evaluación
7. Conclusiones
8. Bibliografía
9. Anexos

1. Introducción:

El presente trabajo práctico integrador tiene como finalidad aplicar de manera práctica los conceptos aprendidos durante la cursada, desarrollando un proyecto en lenguaje Python

El sistema desarrollado consiste en una aplicación de consola que permite gestionar información sobre distintos países, guardando y procesando datos como nombre, población, superficie y continente. A través de una menú interactivo, el usuario puede realizar operaciones de carga, búsqueda, filtrado, ordenamiento y análisis estadístico de la información

Este proyecto busca consolidar los conocimientos técnicos sobre estructuras de datos, modularización, manejo de archivos y persistencia de datos/información, integrando los contenidos trabajados a lo largo del cuatrimestre

2. Objetivos

Objetivo general:

Desarrollar una aplicación de Python que permite gestionar datos de países mediante estructuras de datos adecuadas, archivos, funciones, aplicando los principios de modularización y control de flujo

Objetivos específicos:

- Implementar listas y diccionarios para almacenar y manipular información
- Aplicar funciones y condicionales para organizar el código en bloques claros y reutilizables
- Utilizar archivos CSV para garantizar la persistencia de datos
- Incorporar filtros, búsquedas y ordenamientos según distintos criterios
- Calcular estadísticas básicas sobre los datos almacenados
- Documentar el desarrollo y resultados obtenidos en un informe estructurado

3. Marco teórico:

3.1. Listas:

En Python, una lista es una estructura de datos ordenada que permite almacenar varios elementos en un solo conjunto ordenado.

Se define utilizando corchetes " [] " y puede contener datos de diferentes tipos, como números, textos, diccionarios etc.

En este proyecto, usamos la lista principal (países) para guardar todos los países cargados desde el archivo CSV, esto manera nos permite, filtrarlos, recorrerlos y modificarlos con facilidad

3.2. Diccionarios:

Los diccionarios son estructuras que almacenan pares clave-valor, lo que permite acceder a los datos de forma directa

En este trabajo cada país del sistema se presenta como un diccionario donde las claves son los nombres de los atributos, ("nombre", "población", "superficie", "continente") y los valores son los datos correspondientes a cada uno

De esta manera podemos acceder fácilmente a la información de cada país mediante su clave, este tipo de estructura facilita la organización y manipulación de la información, y permite búsquedas rápidas dentro de la lista general de países.

3.3. Funciones:

Las funciones son bloques de código que nos permiten dividir el programa en secciones más claras y estas pueden ser reutilizables.

Cada función cumple una tarea específica, como agregar un país , buscarlo o mostrar las estadísticas.

Esto mejora la legibilidad, facilita las pruebas del programa y nos ayuda a repetir código, ya que podemos llamar a una función específica previamente definida en cualquier momento

En el proyecto se utilizan funciones para:

- Cargar y guardar datos (obtener_paises, guardar_paises)
- Validar entradas (leer_entero_positivo, leer_entero_no_negativo, leer_texto_no_vacio)
- Operaciones principales (agregar_pais, actualizar_pais, buscar_pais, filtrar_paises, ordenar_paises, mostrar_estadisticas).
- Apoyo y formato (formatear_num, normalizar, mostrar_pais)

3.4. Condicionales:

Los condicionales (if, elif, else) permiten ejecutar diferentes partes del código dependiendo de si se cumple o no una condición lógica.

En este proyecto los usaremos para validar datos ingresados por el usuario, controlar opciones del menú y comparar valores, como, por ejemplo, determinar cuál es el país con mayor o menos población o la existencia del mismo antes de agregarlo.

El uso de la estructura match/case simplifica la selección de opciones dentro del menú interactivo

3.5. Bucles:

Los bucles permiten repetir acciones de forma automática, por ejemplo, podemos recorrer la lista de países o mostrar el menú varias veces.

Nosotros aplicaremos dos bucles diferentes, el bucle `while`, que se usa para mantener activo el menú hasta que el usuario elija salir, y `for`, que se usa para recorrer listas o diccionarios, por ejemplo, cuando necesitar buscar un país dentro de la lista países, filtrar, ordenar y calcular estadísticas.

3.6. Ordenamientos:

El ordenamiento sirve para organizar los datos según un criterio, ya sea por nombre, población o superficie.

Python nos permite hacerlo fácilmente con su función nativa `sorted()` o con el método `.sort()`.

En este proyecto utilizamos la función `sorted()` junto con funciones clave (`clave_nombre`, `clave_poblacion`, `clave_superficie`) para mostrar los países ordenados de forma ascendente o descendente según el campo elegido

3.7. Estadísticas básicas:

Las estadísticas son operaciones que nos permiten analizar los datos, obteniendo los máximos, los mínimos o los promedios. En este trabajo, por ejemplo, se calcula el país con mayor y menor población y los promedios generales de población y superficie.

Estas operaciones se realizan recorriendo la lista y sumando los valores numéricos

La función `mostrar_estadisticas` utiliza un recorrido manual con `for` y comparaciones para encontrar el país con mayor/menor población. También calcula la suma y el promedio de población/superficie, y el conteo de países por continente, utilizando diccionarios como contadores.

3.8. Archivos CSV:

Los archivos CSV permiten almacenar y transferir datos de forma estructurada, donde cada línea representa un registro y los campos están separados por comas.

Python incluye el módulo `csv`, que permite leer y escribir estos tipos de archivos fácilmente, este se emplea mediante.

- `csv.DictReader`: para leer cada fila como un diccionario
- `csv.DictWriter`: para escribir los países en el archivo con encabezados

En este proyecto lo utilizamos para guardar la lista de países y mantener los datos entre ejecuciones del programa.

4. Diseño previo al código

Antes de implementar el programa se definió:

- La estructura de datos: lista de diccionarios
- Las operaciones principales según la consigna (agregar, actualizar, buscar, filtrar, ordenar, estadísticas)
- El uso de un archivo CSV como fuente y destino de datos
- La organización en funciones separadas para lectura/escritura, validaciones, operaciones y menú principal.

4.1. Diagrama de flujo general del programa

En el siguiente diagrama, realizado con ASCIIFlow, se representa el flujo lógico del sistema de gestión de países.

El programa comienza cargando los datos previamente guardados en un archivo CSV (países.csv), si el archivo no existe, se crea uno vacío con los encabezados correspondientes, si existe, se leen y validan los registros para formar la lista inicial de países en memoria.

Luego muestra el menú principal que permite al usuario realizar distintas operaciones, hasta que elija la opción “Guardar y salir”. Cada iteración del menú permite elegir una acción específica entre las siguientes:

- 1. Agregar país: solicita los datos del nuevo país, valida las entradas, evita duplicados y lo agrega a la lista.
- 2. Actualizar país: permite buscar un país por nombre parcial, seleccionarlo y modificar su población o superficie.
- 3. Buscar país: realiza coincidencias parciales por nombre y muestra los resultados encontrados.
- 4. Filtrar países: muestra países filtrados por continente (sin distinguir tildes o mayúsculas) o por rangos de población/superficie.
- 5. Ordenar países: ordena la lista según nombre, población o superficie, en forma ascendente o descendente.
- 6. Mostrar estadísticas: calcula y muestra el país con mayor y menor población, los promedios de población y superficie, y la cantidad de países por continente.
- 7. Guardar y salir: guarda todos los cambios en el archivo países.csv, mostrando un mensaje de cierre y finalizando la ejecución.

De esta forma, el flujo garantiza un proceso ordenado de carga, gestión, análisis y persistencia de datos, manteniendo una estructura modular, clara y fácil de mantener.

```

+-----+
| INICIO |
+-----+
|
| v
+-----+
| Cargar datos desde CSV (obtener_paises) |
| - Si no existe: crear paises.csv vacío |
| - Si existe: leer y validar filas |
+-----+
|
| v
+-----+
| Bucle del menú principal |
| (se repite hasta elegir "Guardar y salir") |
+-----+
|
| v
+-----+
| Mostrar menú y leer opción |
+-----+
|
| v
+-----+
| ¿Opción seleccionada? |
+-----+
|
| +--> [1] Agregar país
| | - Validar datos
| | - Evitar duplicados
| | - Agregar a la lista
|
| +--> [2] Actualizar país
| | - Buscar por nombre parcial
| | - Elegir país
| | - Actualizar pob. y sup.
|
| +--> [3] Buscar país
| | - Coincidencia parcial
| | - Mostrar resultados
|
| +--> [4] Filtrar países
| | - Por continente (normalizar)
| | - Por rango de población/superficie
|
| +--> [5] Ordenar países
| | - Por nombre / pob. / sup.
| | - asc o desc
|
| +--> [6] Mostrar estadísticas
| | - Máx / mín población
| | - Promedios
| | - Cant. por continente
|
| +--> [7] Guardar y salir
| | - guardar_paises()
| | - Fin del bucle
|
| v
+-----+
| FIN |
+-----+

```

5.Desarrollo del caso práctico

El desarrollo del proyecto se realizó utilizando el lenguaje Python y el módulo estándar csv para el manejo de archivos

El programa fue dividido en funciones que cumplen tareas específicas, permitiendo una estructura clara y modular

A continuación se describen las principales funciones implementadas

Lectura y escritura de datos

- obtener_paises(): lee el archivo paises.csv, valida el formato de los datos (campos presentes, población y superficie numéricas y mayores a cero) y carga los países válidos en una lista
- guardar_paises(paises): sobrescribe el archivo csv con el contenido actualizado de la lista de paise
- guardar_y_salir(paises): guarda los datos en el archivo para mantener la persistencia

Validaciones

- leer_entero_positivo(), leer_entero_no_negativo(): aseguran el ingese de valores numericos validos
- leer_texto_no_vacio(): evita campos de texto vacíos
- pais_ya_existe(): impide la carga duplicada de países

Funciones auxiliares o secundarias

- normalizar(): permite comparar textos ignorando mayúsculas y tildes
- formatear_num: mejora la presentación de los numero grandes en las estadísticas sumando un punto (.) cada mil
- mostrar_pais() y listar_pais(): unifican el formato de salida

Operaciones del menú

- agregar_pais(paises): agrega un nuevo país con validación completa de datos
- actualizar_pais(paises): permite modificar la población y superficie de un pais seleccionado por busqueda parcial de nombre
- buscar_pais(paises): muestra países que coinciden parcial o totalmente con el texto ingresado
- filtrar_paises(paises): permite filtrar por continente (normalizado) o por rangos de poblacion/superficie
- ordenar_paises(paises): ordena la lista según la opción elegida y el sentido (ascendente/descendente)
- mostrar_estadisticas(paises): calcula y muestra los valores máximos, mínimo y promedios
- mostrar_menu() y main(): gestiona el flujo principal del programa usando un bucle y la estructura match para opciones

Captura de imágenes del programa “Gestión de países” funcionando:

Menú:

```
PS C:\Users\gasti\OneDrive\Documentos\GitHub\TPI-Programacion> & C:/Users/gasti/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/gasti/OneDrive/Documentos/GitHub/TPI-Programacion/Trabajo Practico Integrador.py"

*****
SISTEMA DE GESTIÓN DE PAÍSES (TPI - Programación 1)
*****
1) Agregar país
2) Actualizar datos de un país
3) Buscar país por nombre
4) Filtrar países
5) Ordenar países
6) Mostrar estadísticas
7) Guardar y salir
*****
Seleccione una opción:
```

Agregar país:

```
PS C:\Users\gasti\OneDrive\Documentos\GitHub\TPI-Programacion> & C:/Users/gasti/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/gasti/OneDrive/Documentos/GitHub/TPI-Programacion/Trabajo Practico Integrador.py"

*****
SISTEMA DE GESTIÓN DE PAÍSES (TPI - Programación 1)
*****
1) Agregar país
2) Actualizar datos de un país
3) Buscar país por nombre
4) Filtrar países
5) Ordenar países
6) Mostrar estadísticas
7) Guardar y salir
*****
Seleccione una opción: 1

--- Bienvenido! Por favor, ingrese un pais---
Nombre: estonia
Poblacion: 1372000000
Superficie (Km2): 45335
continente: europa
El pais estonia fue agregado correctamente.
```


Filtrar países: (por continente, rango de población, rango de superficie)

```
*****
SISTEMA DE GESTIÓN DE PAÍSES (TPI - Programación 1)
*****
1) Agregar país
2) Actualizar datos de un país
3) Buscar país por nombre
4) Filtrar países
5) Ordenar países
6) Mostrar estadísticas
7) Guardar y salir
*****
Seleccione una opción: 4

--- Filtrar países ---
1) Por continente
2) Por rango de población
3) Por rango de superficie
Seleccione una opción: 1
Ingrese el continente: america
Se encontraron 24 país(es):
- Argentina | Población: 4522220 | Superficie: 27000000 km2 | Continente: América
- Brasil | Población: 213993437 | Superficie: 8515767 km2 | Continente: América
- Chile | Población: 19116209 | Superficie: 756096 km2 | Continente: América
- Uruguay | Población: 3426260 | Superficie: 176215 km2 | Continente: América
```

Mostrar estadísticas:

```
*****
SISTEMA DE GESTIÓN DE PAÍSES (TPI - Programación 1)
*****
1) Agregar país
2) Actualizar datos de un país
3) Buscar país por nombre
4) Filtrar países
5) Ordenar países
6) Mostrar estadísticas
7) Guardar y salir
*****
Seleccione una opción: 6

=== ESTADÍSTICAS ===
• País con MAYOR población : China (1.411.778.724)
• País con MENOR población : Nauru (10.834)
• Promedio de población : 80.786.946
• Promedio de superficie : 1.348.727

• Cantidad de países por continente:
  - América: 24
  - Europa: 21
  - Asia: 26
  - Oceanía: 12
  - África: 20
  - asia: 1
```

6. Resultados y evaluación

Durante las pruebas realizadas, el sistema permitió:

- Cargar nuevos países con validaciones que evitan campos vacíos o datos no numéricos.
- Actualizar correctamente los datos de población y superficie.
- Buscar países por coincidencias parciales de nombre.
- Filtrar por continente (ignorando mayúsculas y tildes) y por rangos válidos de población y superficie.
- Ordenar la información según los criterios requeridos sin errores.
- Calcular estadísticas coherentes con los datos cargados.

Entre los principales aciertos que tuvimos, se destaca el modularización del código, el uso de funciones auxiliares para validación y formato y la protección a errores comunes (como filas inválidas en el CSV o entradas incorrectas por parte del usuario)

Como dificultad principal, podríamos decir que fue el manejo correcto de tildes y formatos en los datos del CSV, lo cual lo pudimos resolver implementando la función `normaliza` y el módulo estándar `unicodedata` y agregando controles adicionales sobre los campos numéricos

En conjunto, el proceso de desarrollo no fue trivial: requirió corregir versiones iniciales, ajustar nombres de funciones, repensar el flujo de guardado y estudiar soluciones para problemas específicos como el manejo de acentos y el formateo del CSV. Estos desafíos contribuyeron a una mejor comprensión práctica de los contenidos trabajados en la materia.

La colaboración permitió revisar código en equipo, comparar estilos de validación, discutir nombres de funciones y decidir la estructura final.

Cada corrección implicó un aprendizaje adicional sobre buenas prácticas de Python, especialmente sobre la importancia de probar, comentar y documentar el código.

7. Conclusión

En conclusión, el desarrollo del sistema de gestión de países nos permitió aplicar de forma práctica los conceptos aprendidos a lo largo del cuatrimestre en la materia Programación 1, combinando estructuras de datos, funciones, validaciones y manejo de archivos CSV.

Al trabajar en conjunto a lo largo del proyecto aprendimos a planificar, dividir tareas, corregir y anticipar posibles errores, fortaleciendo la comprensión del flujo lógico de un programa y la importancia de la organización del código. A su vez pudimos revisar, resolver y mejorar las distintas partes del programa, logrando las metas propuestas. El sistema funciona, es claro, está comentado, está bien estructurado y tiene buena modularización .

En síntesis, el proyecto nos permitió consolidar los conocimientos adquiridos, demostrando la capacidad de transformar los contenidos teóricos en una solución práctica y funcional.

8. Bibliografía:

- Python Software Foundation. (2024). Python Documentation: Data Structures and CSV module. <https://docs.python.org/3/>
- Real Python. (2023). Python Lists, Dictionaries, and File Handling Tutorials. <https://realpython.com>
- W3Schools. (2024). Python Tutorial: Lists, Dictionaries, and File Handling. <https://www.w3schools.com/python/>
- Apuntes de la cátedra Programación 1- UTN

9. Anexos

Enlace al video subido a YouTube:

[Sistema de Gestion de paises \(TPI Programacion I UTN\)](#)

El video muestra el funcionamiento del sistema en ejecución, incluyendo ejemplos de carga, búsqueda, filtrado, ordenamiento y cálculo de estadísticas.