TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

Trabajo Practico Integrador: Programación I

- Alumnos: Cristian Alejandro Tapia (c13), Daniela Velazquez (c12)
- Materia: Programación I
- Coordinador: Alberto Cortez
- Profesor: Cinthia Rigoni, Sebastian Bruselario, Ariel Enferrel
- Tutores: Franco Gonzalez, Miguel Barrera

Índice

Introducción:

El objetivo principal de este trabajo fue desarrollar una aplicación de consola en Python que permita gestionar información sobre países.

El foco estuvo en afianzar el uso de estructuras de datos (listas y diccionarios), la modularización con funciones, y la implementación de técnicas de filtrado, ordenamiento y cálculo de estadísticas básicas. El sistema debe ser capaz de leer datos desde un archivo CSV, realizar consultas y generar indicadores clave a partir del dataset.

Marco Teórico

En este apartado explicamos los conceptos fundamentales de Programación utilizados para el desarrollo del sistema.

Estructuras de Datos: Listas y Diccionarios

Un **diccionario** en Python es una colección desordenada, mutable e indexada por claves, que se utiliza para almacenar datos en pares de clave: valor.

- Aplicación en el TPI: Cada país se modeló como un diccionario. Esto permitió acceder a los atributos del país de forma clara y semántica (ej. pais ["POBLACION"]), reflejando la estructura del CSV (nombre, poblacion, superficie, continente) .

Las **listas** son colecciones ordenadas y mutables de elementos. En este proyecto, la lista fue la estructura central del dataset.

- **Aplicación en el TPI:** La lista principal (paises) almacenó el conjunto total de diccionarios, representando la colección completa de países. Esto facilitó la iteración sobre el dataset para aplicar búsquedas, filtros y ordenamientos.

Modularización y Funciones

La modularización es la práctica de dividir el código en funciones con responsabilidades únicas.

- **Aplicación en el TPI:** Se cumplió con el principio de "una función = una responsabilidad". Por ejemplo, las funciones de Persistencia (ObtenerPaises, PersistirCsv) solo manejan el archivo, mientras que las de Lógica de Negocio (AgregarPais, ActualizarDatos) sólo aplican las reglas del sistema.

Ordenamiento

Para ordenar los países por diversos criterios (Nombre, Población, Superficie) se utilizó la función nativa sorted().

- **Función sorted() con key**: Esta función devuelve una nueva lista ordenada a partir de una iterable. El uso crucial fue el parámetro key, que permite especificar una función para determinar qué valor de cada elemento debe usarse como criterio de ordenación (por ejemplo, el valor de la clave "POBLACION" del diccionario). El argumento reverse=True/False permitió gestionar el orden ascendente o descendente.

Estadísticas y Agrupación

Funciones min() y max()

Estas funciones son utilizadas para encontrar el elemento más pequeño o más grande de un iterable.

- Aplicación en el TPI: Se utilizaron con el parámetro key para encontrar eficientemente el país completo (el diccionario) con la mayor y menor población.

Agrupación (setdefault)

Para calcular indicadores de agrupación, como la cantidad de países por continente, se utilizó la función setdefault() de los diccionarios.

- Función setdefault(): Permite obtener el valor de una clave, y si la clave no existe, la inserta con un valor por defecto (en este caso, una nueva lista vacía) antes de devolver dicho valor. Esto es fundamental para construir el diccionario de agrupación: {continente: [país1, país2, ...]}.

Archivos CSV

El acceso a los datos se realiza a través de un archivo en formato **CSV** (Comma Separated Values).

- **Aplicación en el TPI:** El módulo de persistencia (ObtenerPaises y PersistirCsv) se encarga de: 1) Leer el archivo y convertir cada fila en un diccionario. 2) Reescribir la lista completa de diccionarios al archivo cada vez que se produce una modificación.

Caso Práctico

Estructura de Diseño Modular

El diseño implementado sigue una arquitectura simple de tres capas, que separa el flujo de control, la lógica del negocio y la interacción con los datos.

En la siguiente página podremos visualizar el cuadro.

Capa	Funciones	Responsabilidad Principal
Programa principal	programa_principal()	Actúa como el motor del sistema, presentando el menú y delegando la ejecución.
Lógica de Negocio	AgregarPais(), ActualizarDatos(), BuscarPais(), OrdenarPaises()	Implementa las reglas del sistema, coordinando las Utilidades y la Persistencia.
Capas de datos y Utilidades	ObtenerPaises(), PedirNombre(), PedirCantidad()	Maneja la lectura/escritura del CSV y garantiza la calidad (validación) de los datos ingresados por el usuario.

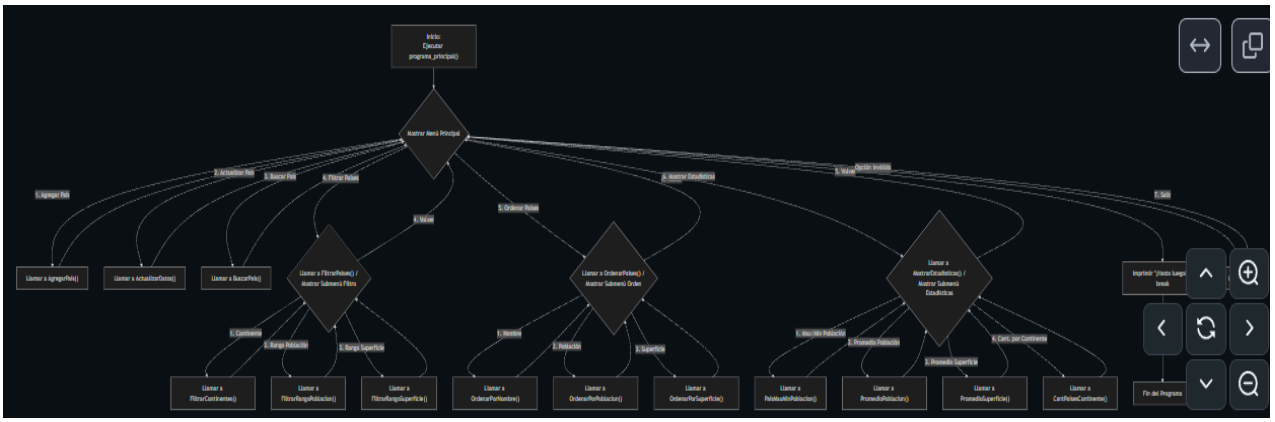
Diagrama de Flujo del Programa

El siguiente diagrama visualiza la modularización y el flujo de operaciones principales del sistema: (fragmento del código)

```
graph TD

    A["Inicio:<br>Ejecutar programa_principal()"] --> B{"Mostrar Menú Principal"};

    B -->|1. Agregar País| C["Llamar a AgregarPais()"];

    C --> B;
```

TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

```
B -->|2. Actualizar País| D["Llamar a ActualizarDatos()"];

D --> B;

B -->|3. Buscar País| E["Llamar a BuscarPais()"];

E --> B;

B -->|4. Filtrar Países| F{"Llamar a FiltrarPaises() /<br>Mostrar Submenú Filtro"};

F -->|1. Continente| F1["Llamar a FiltrarContinentes()"];

F1 --> F;

F -->|2. Rango Población| F2["Llamar a<br>FiltrarRangoPoblacion()"];

F2 --> F;

F -->|3. Rango Superficie| F3["Llamar a<br>FiltrarRangoSuperficie()"];

F3 --> F;

F -->|4. Volver| B;
```



Estrategia de Validaciones y Manejo de Errores

El código incluye un robusto sistema de validación para garantizar la integridad de los datos, tal como lo requiere la consigna.

- **Control de Errores en CSV:** La función ObtenerPaises() implementa bloques try-except para capturar posibles errores de formato (ej. valores no numéricos en Población o Superficie), evitando que el programa se detenga al leer datos corruptos.

- **Validación de Entradas del Usuario:** Funciones como PedirCantidad() utilizan bucles while True para asegurar que el usuario ingrese únicamente números enteros positivos. Las funciones de validación también impiden campos vacíos.

- **Manejo de Búsquedas**: Se implementaron mensajes claros de error o éxito para evitar fallos al ingresar filtros inválidos o búsquedas sin resultados, mejorando la experiencia del usuario.

Resultados Obtenidos y Conclusiones

Funcionalidades Implementadas

El sistema cumple con la totalidad de las funcionalidades mínimas requeridas :

- **Gestión de Datos:** Agregar un país con validación de campos vacíos y Actualizar los datos de Población y Superficie de un país existente.

- **Consultas:** Buscar un país por nombre (coincidencia parcial o exacta).

- **Filtros:** Filtrar por Continente , Rango de Población y Rango de Superficie.

- **Ordenamientos:** Ordenar la lista de países por Nombre , Población y Superficie, en orden ascendente o descendente.

- **Estadísticas:** Mostrar país con mayor y menor población , Promedio de población y superficie , y Cantidad de países por continente.

Conclusión

El desarrollo de este TPI nos permitió afianzar y aplicar conceptos teóricos clave.

- **Aprendizaje Central:** La **modularización** demostró ser la herramienta más valiosa, ya que la división del código en funciones de responsabilidad única simplificó tanto el desarrollo como el proceso de prueba y corrección de errores. Por ejemplo, el módulo de **Validaciones** permitió aislar y testear la lógica de control de entradas de manera independiente.

- **Uso de Estructuras:** El uso combinado de la **Lista de Diccionarios** fue clave para modelar un dataset relacional simple, demostrando la versatilidad de estas estructuras para trabajar con datos estructurados en Python.

- **Organización del Equipo:** La comunicación constante y la asignación clara de tareas permitió un desarrollo eficiente y organizado, facilitado por el uso de Git para la gestión de versiones y la integración del trabajo.

Referencias y Repositorio

Fuentes Bibliográficas

- Fuente de funcion sorted(): https://docs.python.org/3/howto/sorting.html#ascending-and-descending

- Fuente de funciones Min() - Max(): https://realpython.com/python-min-and-max

- Fuente de setdefault() : https://www.w3schools.com/python/ref_dictionary_setdefault.asp

Repositorio GitHub

- **Enlace:** https://github.com/CTapia10/TPI-Programacion/tree/main