# Informe Teórico de la Aplicación

## 1. Introducción

El presente informe detalla el diseño teórico de una aplicación en Python, desarrollada como parte del Trabajo Práctico Integrador (TPI) de la materia Programación 1.

El objetivo principal del sistema es desarrollar una aplicación de consola que permita gestionar información sobre países. El sistema es capaz de leer datos desde un archivo CSV, almacenar esta información en memoria y ofrecer al usuario un menú de opciones para realizar consultas, aplicar filtros, ordenar los datos y generar indicadores estadísticos clave a partir del dataset.

La aplicación gestionará una colección de países, donde cada país se define por su **Nombre** (string), **Población** (int), **Superficie** (int) y **Continente** (string).

## 2. Diseño y Flujo de la Aplicación

### 2.1. Flujo de Operaciones Principal

El flujo de la aplicación se centra en un menú de consola interactivo. A continuación, se detalla el esquema de operaciones basado en los archivos main.py y las funciones importadas:

1. **Inicio:** La aplicación se inicializa.
2. **Carga de Datos:** El sistema llama a leer\_csv() y carga los datos de Paises.csv en la estructura de datos principal: un **diccionario** donde la clave es el nombre del país y el valor es un diccionario con todos sus datos (variable diccionario en main.py).
3. **Menú Principal:** Se muestra al usuario un menú de opciones en bucle (while True):
   * **1. Buscar país:** Solicita al usuario un término de búsqueda (dato). Llama a buscar\_pais() (que itera sobre el diccionario principal) y muestra los resultados (una lista de tuplas).
   * **2. Filtrar países:** Muestra un submenú para elegir el criterio (Continente, Población, Superficie). Pide los datos necesarios y llama a filtrar\_paises() (que itera y filtra el diccionario principal), mostrando los resultados.
   * **3. Ordenar países:** Muestra un submenú para elegir el criterio (Nombre, Población, Superficie) y el tipo de orden (Ascendente/Descendente). Llama a ordenar\_paises() (que convierte el diccionario a lista y la ordena), mostrando los resultados.
   * **4. Mostrar estadísticas:** Llama a mostrar\_estadisticas() para calcular y mostrar un resumen de los datos (Población máx/mín, promedios y conteo por continente).
   * **0. Salir:** Termina la ejecución del programa (break).
4. **Validación:** Tras cada operación, se imprimen los resultados o mensajes de error si la opción no es válida. (Las funcionalidades de "Agregar país" y "Actualizar país" de la consigna aún no están implementadas en este flujo inicial).

## 3. Fundamentos Teóricos Aplicados

Esta sección detalla los conceptos de programación clave que se utilizan como base para el diseño y la lógica de la aplicación.

### 3.1. Listas

Una lista es un conjunto ordenado de valores que se identifican por medio de un índice. Los valores que componen una lista se denominan elementos.

lista\_ejemplo = ["hola", 2.0, 5, [10, 20]]

En este proyecto, una **lista** *no* es la estructura de datos principal que almacena los países. En su lugar, las listas se usan para almacenar los **resultados** de las operaciones. Las funciones buscar\_pais, filtrar\_paises y ordenar\_paises devuelven los datos procesados en forma de lista de tuplas (nombre, datos), que luego es consumida por mostrar\_paises.

### 3.2. Diccionarios

Los diccionarios son tipos de dato compuestos similares a las listas, excepto en que pueden usar como índice cualquier tipo inmutable (como un string).

>>> ing\_a\_esp = {}  
>>> ing\_a\_esp['one'] = 'uno'  
>>> ing\_a\_esp['two'] = 'dos'

Para este proyecto, se usa un **diccionario** como la estructura de datos *principal* (la variable diccionario en main.py). Esta estructura es un **"diccionario de diccionarios"**: la clave del diccionario principal es el nombre del país (ej: "Argentina") y el valor asociado es otro diccionario que contiene todos los datos de esa fila (ej: {'Población': 45376763, ...}). Este diseño permite un acceso O(1) (casi instantáneo) a los datos de un país si se conoce su nombre.

### 3.3. Funciones

Una función es un conjunto breve de instrucciones que permiten alcanzar fácilmente un pequeño objetivo, siendo la base para la reutilización del código y la simplificación del objetivo.

def cuadrado(x):  
 return x \*\* 2

La modularización es un requisito clave del proyecto. Por ello, cada funcionalidad principal del menú (Buscar, Filtrar, Ordenar, Estadísticas) y las tareas auxiliares (Leer CSV, Mostrar Menú) están implementadas como una **función** separada e importadas en main.py, cumpliendo con el principio de responsabilidad única.

### 3.4. Condicionales

Una sentencia condicional es un esquema de instrucciones que permite escoger uno de entre dos caminos lógicos (o varios).

# Se evalúa si es igual a 6 o no  
if num == 6:  
 print("El número recibido es un 6")  
else:  
 print("El número recibido NO es un 6")

Los **condicionales** (como if, elif y else) se usan en todo el proyecto: en main.py para gestionar los submenús; en filtrar\_paises.py para aplicar los filtros (if min\_poblacion is not None...); en ordenar\_paises.py para determinar el criterio (if criterio == "País":...); y en buscar\_pais.py para encontrar coincidencias (if palabra in pais.lower():).

### 3.5. Ordenamientos

El ordenamiento es el proceso de reorganizar los elementos de una colección (como una lista) según un criterio (orden ascendente, descendente).

En esta aplicación, se implementa una estrategia personalizada: la función ordenar\_paises convierte el diccionario de países en una lista de tuplas. Luego, llama a la función ordenar\_lista (que contiene un algoritmo *Bubble Sort* escrito a mano) para reorganizar la lista según el criterio y el sentido elegidos por el usuario.

**Algoritmos comunes:**

* Burbuja (bubble sort)
* Selección (selection sort)
* Inserción (insertion sort)
* Merge Sort
* Quick Sort

### 3.6. Estadísticas básicas

Este concepto se refiere al cálculo de indicadores clave a partir de un conjunto de datos, como la media (promedio), mediana, máximos y mínimos.

La función **mostrar\_estadisticas** implementa esta funcionalidad. Itera sobre el diccionario principal (**.items()**) para acumular la población y superficie total, contar países por continente (usando un diccionario auxiliar), y encontrar los países con población máxima y mínima (**float('inf')**). Finalmente, calcula y muestra los promedios y el conteo por continente.

* **Media**: Suma de todos los valores dividida por el número de valores. Sinónimos: promedio
* **Media ponderada**: Suma de todos los valores multiplicados por cada ponderación y dividida por la suma de las ponderaciones. Sinónimo: promedio ponderado
* **Mediana**: Valor tal que la mitad del número de datos se encuentra por encima y la otra mitad por debajo de dicho valor. Sinónimo: Percentil 50

### 3.7. Archivos CSV

CSV (Valores Separados por Comas) es el formato más común de importación y exportación de hojas de cálculo y bases de datos.

Se utiliza el módulo csv de Python. Específicamente, **leer\_csv.py** usa **csv.DictReader** para leer **Paises.csv** fila por fila. Cada fila (que es un diccionario) se almacena inmediatamente en un diccionario principal (datos) usando el nombre del país (extraído de fila["País"]) como la clave principal.

## 4. Bibliografía

* csv — CSV File Reading and Writing: https://docs.python.org/es/3.13/library/csv.html
* (Andrés Becerra Sandoval) Introducción a la programación con Python: https://ia802802.us.archive.org/34/items/IntroduccinALaProgramacinConPythonBecerra/Introducci%C3%B3n%20a%20la%20programaci%C3%B3n%20con%20Python%20-%20Becerra.pdf
* (Bruce-Gedeck) Estadística Práctica para Ciencia de Datos Con R y Python: https://es.scribd.com/document/660928067/Bruce-Gedeck-Estadistica-Practica-para-Ciencia-de-Datos-con-R-y-Python
* (Andrés Marzal, Isabel Gracia) Introducción a la programación con Python: https://www.google.com/search?q=https://aulavirtual.fio.unam.edu.ar/pluginfile.php/92534/mod\_resource/content/1/Introducci%25C3%25B3n%2520a%2520la%2520Programaci%25C3%25B3n%2520con%2520Python.pdf