

# Informe Teórico Programación

## Introducción

El presente informe desarrolla de manera rigurosa los conceptos teóricos fundamentales utilizados en la elaboración del proyecto de gestión, análisis y almacenamiento de datos de países. Estos conceptos —listas, diccionarios, funciones, condicionales, ordenamientos, estadísticas y manejo de archivos CSV— constituyen la base de la programación estructurada en Python, y permiten modelar, procesar y organizar información de manera eficiente.

El marco teórico se fundamenta en bibliografía especializada, especialmente *Think Python* (Downey, 2015), referente clásico en introducción a la programación, junto con otras fuentes complementarias. Posteriormente, estos conceptos se vinculan detalladamente con su implementación concreta dentro del proyecto

## Diccionarios

Los diccionarios son estructuras que almacenan pares clave–valor, permitiendo acceso rápido a los datos mediante una clave.

Son útiles cuando se requiere representar entidades con atributos, ya que permiten un acceso eficiente y semántico.

McKinney (2017) resalta que los diccionarios son fundamentales para representar estructuras tipo “registro”, similares a filas de una tabla en una base de datos.

Ejemplo:

```
1 diccionario = {  
2     "nombre": "Juan",  
3     "edad": 30,  
4     "ciudad": "Madrid"}
```

## Funciones

Una función es una secuencia nombrada de instrucciones que realiza un

cómputo. Una función permite organizar el código, utilizarlo y mejorar la claridad de un programa.

Permiten modularidad, claridad y reducen la repetición de código (Downey, 2015). Una función puede recibir parámetros y devolver valores, formando parte esencial de la programación estructurada y del diseño basado en abstracciones.

Según Guttag (2016), el uso adecuado de funciones mejora la legibilidad y favorece el mantenimiento de programas extensos.

Ejemplo:



```
1 def sumar(a,b):  
2     return a + b
```

### Condicionales

Los condicionales permiten ejecutar instrucciones dependiendo de si una expresión lógica es verdadera o falsa.

Este mecanismo es fundamental para la toma de decisiones y la gestión del flujo del programa (Downey, 2015).

Las sentencias `if`, `elif` y `else` constituyen la base de estas operaciones.

Ejemplo:



```
1 if edad >= 18:  
2     print("Eres mayor de edad.")  
3 else:  
4     print("Eres menor de edad.")
```

### Estadísticas básicas

Las estadísticas básicas incluyen operaciones fundamentales como sumas, promedios, máximos y mínimos, usando listas y funciones integradas.

Estas permiten sintetizar información numérica de un conjunto de datos. En programación, dichas operaciones pueden implementarse manualmente mediante bucles o usando funciones nativas como `min()`, `max()`, `sum()` y `len()` (Downey, 2015).

Ejemplo:

```
● ● ●  
1  numeros = [1, 2, 3, 4, 5]  
2  prom = sum(numeros) / len(numeros)  
3  print("El promedio es:", prom)
```

## Ordenamientos

El **ordenamiento** consiste en reorganizar elementos siguiendo un criterio determinado. Python incorpora funciones como `sort()` y `sorted()`, las cuales permiten ordenar secuencias utilizando parámetros como `key` o `reverse` (Downey, 2015).

Según McKinney (2017), el ordenamiento es esencial para la organización de datos, permitiendo comparaciones, búsquedas eficientes y análisis descriptivos.

## Listas

En Python, una **lista** es una estructura de datos secuencial, ordenada y **mutable**, capaz de almacenar múltiples elementos en posiciones indexadas (Downey, 2015). Su mutabilidad permite insertar, eliminar o modificar elementos en tiempo de ejecución, convirtiéndolas en una estructura ideal para colecciones dinámicas.

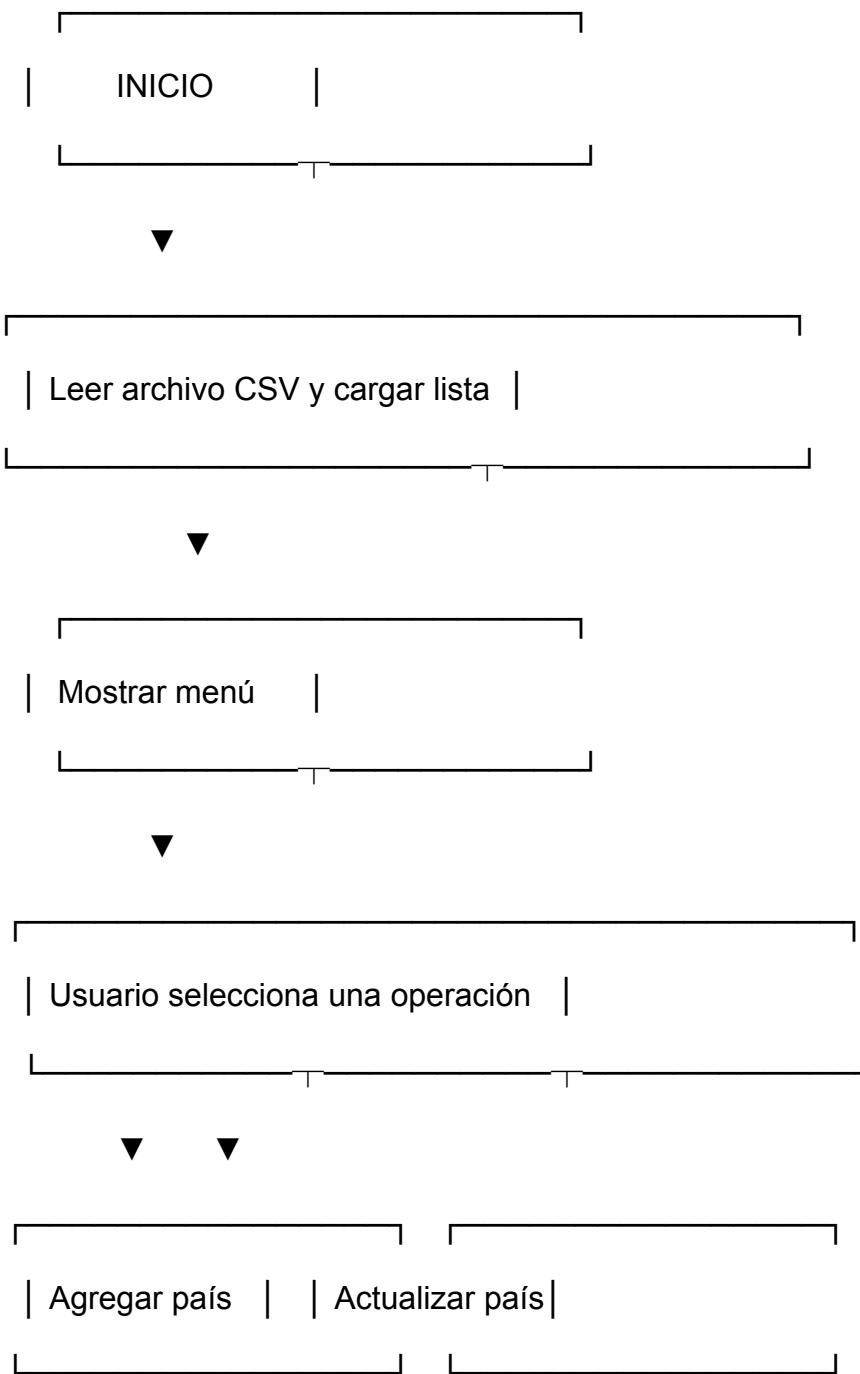
Guttag (2016) destaca que las listas permiten representar conjuntos de datos heterogéneos y son una herramienta flexible para implementar algoritmos de búsqueda, ordenamiento o filtrado.

## Archivos CSV

Los **archivos CSV (Comma-Separated Values)** son archivos de texto que representan tablas de datos mediante filas separadas por saltos de línea y columnas separadas por comas. Este formato es ampliamente utilizado por su simplicidad y compatibilidad (McKinney, 2017).

En Python, pueden manipularse mediante el módulo estándar csv o mediante operaciones manuales utilizando open(), split() y readlines().

#### 4. Diagrama del Flujo de Operaciones



▼ ▼

| Buscar / Filtrar / |

| Ordenar / Estadísticas |

▼

| Mostrar resultados |

▼

| FIN |

## Conclusión

Los conceptos teóricos desarrollados constituyen el fundamento del diseño e implementación del proyecto. La estructura de datos (listas y diccionarios), la modularidad (funciones), el control de flujo (condicionales), las capacidades analíticas (estadísticas y ordenamientos) y la persistencia mediante CSV permiten construir una aplicación completa, organizada y extensible.

El proyecto implementa fielmente las metodologías propuestas en la bibliografía, mostrando un dominio adecuado de la programación estructurada en Python.

---

## 6. Bibliografía

- Downey, A. (2015). *Think Python: How to Think Like a Computer Scientist*. Green Tea Press.
- Guttag, J. (2016). *Introduction to Computation and Programming Using Python*. MIT Press.
- McKinney, W. (2017). *Python for Data Analysis*. O'Reilly Media.
- Python Software Foundation. (2024). *The Python Tutorial*.  
<https://docs.python.org>