5. INFORME DE ANÁLISIS Y CONCLUSIONES DEL TALLER

Algoritmo: Búsqueda Lineal (O(n))

Estudiante: Jhoan Manuel Chavez Ocampo

---

1. Objetivo y Tareas Realizadas

El objetivo de este taller fue implementar el algoritmo de **Búsqueda Lineal** en las funciones de un sistema de gestión de tienda de electrónica.

En el archivo ejercicios_practicos.py, logré implementar correctamente:

1. **Búsqueda Básica:** Encontrar la posición (índice) de un elemento simple.
2. **Filtros Complejos:** Buscar productos por **nombre**, **marca**, por **rango de precio** y por **disponibilidad y stock** (usando la lógica AND).
3. **Estadísticas:** Recorrer las listas para obtener el valor total de inventario y el conteo de productos por categoría.
4. **Análisis de Rendimiento:** Modifiqué el algoritmo para contar el número de **comparaciones** realizadas.

---

2. Análisis de la Complejidad Temporal

La Búsqueda Lineal tiene una **complejidad de O(n)** (Orden de n), donde n es el número de elementos en la lista.

Escenario	Comparaciones Necesarias	Conclusión

Mejor Caso	**1**	El elemento está al inicio de la lista. Rápido (O(1)).
Peor Caso	***n***	El elemento está al final de la lista o no existe. Lento, ya que se revisa la lista entera.

Reflexión del Estudiante: Para las tareas de la tienda que requieren un recorrido completo (como calcular el inventario total o contar todas las marcas), la complejidad **siempre es O(n)** porque debemos tocar todos los elementos.

---

3. Conclusiones sobre la Aplicación

¿Cuándo es eficiente en la Tienda?

La Búsqueda Lineal es suficiente y eficiente para este sistema porque:

1. **Es simple y fácil de codificar.**
2. **Los datos son pequeños:** Con menos de 50 productos y 10 empleados, la diferencia de velocidad con un algoritmo más complejo es mínima.
3. **Los datos no están ordenados:** La Búsqueda Lineal no requiere que la lista esté ordenada, lo que ahorra tiempo de pre-procesamiento.

¿Cuál es su principal limitación?

Si la tienda creciera y tuviera **millones** de productos (escala grande), la Búsqueda Lineal (O(n)) sería muy lenta.

En ese caso, para búsquedas rápidas por ID, se necesitaría ordenar la lista y usar **Búsqueda Binaria (O(log n))**.