

Caso de Estudio: Sistemas de Búsqueda para una Biblioteca Digital Estudiantil

Contexto

La Universidad desea desarrollar un prototipo inicial de un sistema de búsqueda para su futura **Biblioteca Digital Estudiantil**. Este sistema permitirá localizar información de forma rápida y eficiente dentro de una colección de datos como libros, autores, categorías y códigos.

Como parte del proceso formativo, se asigna a los estudiantes la tarea de investigar, diseñar e implementar **algoritmos de búsqueda básicos en C#**, los cuales servirán como base para el proyecto final. Esta actividad pretende que los estudiantes comprendan cómo funcionan estos métodos dentro de sistemas reales, tal como los que usan las bibliotecas modernas.

Objetivo del Caso de Estudio

Aplicar conceptos de búsqueda en estructuras de datos mediante algoritmos manuales implementados en C#. Además, se busca fortalecer el trabajo colaborativo, el análisis técnico y la documentación correcta según las **normas APA 7ma edición**.

Desarrollo de la Investigación Teórica

¿Qué es un algoritmo de búsqueda?

Un **algoritmo de búsqueda** es un procedimiento que permite localizar un elemento específico dentro de una estructura de datos. Su propósito es determinar si el elemento existe y, de ser así, obtener su posición. Estos algoritmos son esenciales en computación, ya que optimizan el acceso a la información en listas, arreglos, bases de datos y sistemas digitales.

Tipos de Algoritmos de Búsqueda

1. Búsqueda Lineal

La búsqueda lineal revisa cada elemento uno por uno desde el inicio hasta encontrar el valor buscado o llegar al final.

Es el algoritmo más simple, pero también el más lento cuando los conjuntos de datos son grandes.

Ventajas:

- Fácil de implementar.
- Funciona con datos no ordenados.

Desventajas:

- Ineficiente para listas extensas.
- Tiempo de búsqueda proporcional al tamaño de la lista.

2. Búsqueda Binaria

La búsqueda binaria divide repetidamente la lista a la mitad para encontrar un elemento. Requiere que los datos estén **ordenados**.

Ventajas:

- Mucho más rápida que la búsqueda lineal.
- Ideal para listas grandes ordenadas.

Desventajas:

- No funciona con datos desordenados.
- Su implementación es un poco más compleja.

3. Búsqueda en Estructuras No Lineales

Se aplica en estructuras como **árboles**, **grafos** o índices avanzados.

Algunas variantes comunes son:

- **Búsqueda en profundidad (DFS)**
- **Búsqueda por amplitud (BFS)**
- **Búsqueda en árboles binarios de búsqueda (BST)**

Ventajas:

- Muy eficientes para grandes volúmenes de datos.
- Permiten organizar la información de forma jerárquica.

Desventajas:

- Requieren estructuras más complejas.
- Su mantenimiento e implementación pueden demandar más recursos.

Casos de Uso Reales

Estos algoritmos se utilizan en una enorme cantidad de sistemas, por ejemplo:

Bibliotecas digitales: localizar libros, autores o categorías.

Motores de búsqueda (Google, Bing): encontrar coincidencias entre millones de datos.

Bases de datos: consultas SQL optimizadas mediante índices.

Sistemas de recomendación: buscar elementos similares al comportamiento del usuario.

Aplicaciones móviles: filtrado en listas, catálogos o inventarios.

Ejemplos en el Mundo Profesional

Los sistemas de gestión de bases de datos como **SQL Server, MySQL o PostgreSQL** aplican búsquedas indexadas (árboles B+).

Los motores de inteligencia artificial utilizan búsquedas heurísticas para tomar decisiones rápidas.

Plataformas como Amazon o Netflix usan algoritmos de búsqueda para filtrar y recomendar contenido.

Los navegadores usan algoritmos de búsqueda para autocompletar texto y sugerencias.

Ventajas y Desventajas por Técnica

Técnica	Ventajas	Desventajas
Búsqueda Lineal	Simple, funciona con datos desordenados	Lenta en listas grandes
Búsqueda Binaria	Muy rápida, eficiente	Requiere datos ordenados
Búsqueda en Árboles/Grafos	Escalable, profesional	Implementación compleja

Referencias (APA 7ma Ed.)

Aquí te dejo unas referencias que podés usar sin clavo:

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.

Goodrich, M., Tamassia, R., & Goldwasser, M. (2014). *Data Structures and Algorithms in C#*. Wiley.

Sedgewick, R., & Wayne, K. (2011). *Algorithms* (4th ed.). Addison-Wesley.

Microsoft. (2023). *C# documentation*. Microsoft Learn. <https://learn.microsoft.com>

Knuth, D. E. (1998). *The Art of Computer Programming: Sorting and Searching* (Vol. 3). Addison-Wesley.