Búsqueda y Ordenamiento en algoritmos de Python

Alejo Tomas Oliva Coca - alejooliva070@gmail.com

Christian Fernando Ormachea - ormacheachristianf@gmail.com

Programación 1

Profesor/a: Ing. Laura Fernández

Fecha de entrega: 2 de junio de 2025

Índice

1 – Introducción

La búsqueda y el ordenamiento son las bases para el rápido funcionamiento de las aplicaciones más cotidianas, gracias a los algoritmos de búsqueda y ordenamiento somos capaces de visibilizar a través de la búsqueda grandes cantidades de contenido de manera ordenada en Netflix, Gmail o YouTube.

La búsqueda y el ordenamiento son un conjunto de algoritmos conocidos que ayudan a buscar y organizar los datos de la manera más eficiente dependiendo de los casos de uso que estemos buscando implementar, gracias a esta variedad de soluciones hoy en día sabemos que conviene hacer en caso de que se nos presente alguna de estas problemáticas.

2 – Marco Teórico

Búsqueda

La búsqueda es una operación fundamental en programación que se utiliza para encontrar un elemento específico dentro de un conjunto de datos. Es muy común en bases de datos, sistemas de archivos y algoritmos de inteligencia artificial, hay distintos tipos de búsqueda:

- *Búsqueda lineal:* Es el tipo más común de búsqueda, consiste en recorrer cada elemento del conjunto de datos de forma secuencial hasta encontrar el elemento deseado.

- *Búsqueda binaria:* Este tipo de búsqueda funciona solamente en conjuntos de datos ordenados, si se intenta usar en un conjunto/lista de datos no ordenados el resultado puede ser incorrecto. El método consiste en dividir el conjunto de datos en dos mitades y busca el elemento deseado en la mitad correspondiente. Repite este proceso hasta encontrar el elemento o determinar que no está en el conjunto de datos.

- *Búsqueda de Interpolación*: Este algoritmo es una mejora del algoritmo de búsqueda binaria porque estima la posición del elemento deseado en función de su valor.

• *Búsqueda hash*: Es un algoritmo de búsqueda que utiliza una función hash para asignar cada elemento a una ubicación única en una tabla hash. Esto permite acceder a los elementos en tiempo constante, lo que lo hace muy eficiente para conjuntos de datos grandes.

Importancia de la búsqueda:

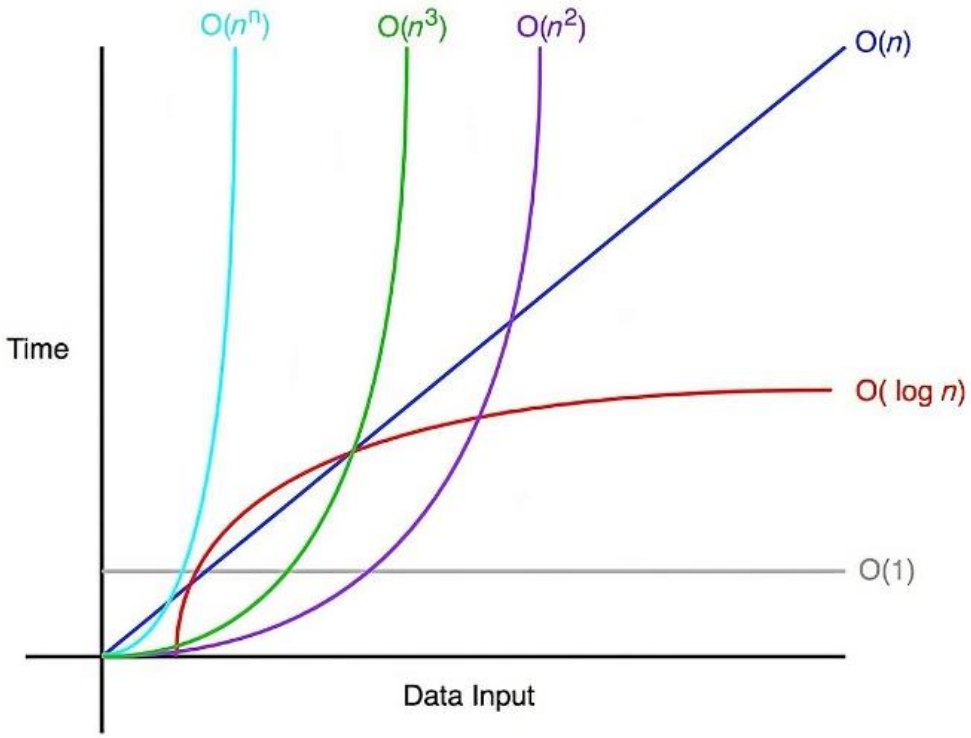
La búsqueda es importante en programación porque se utiliza en una amplia variedad de aplicaciones, se utiliza en algoritmos de búsqueda para encontrar palabras clave en un documento o para para encontrar un archivo con un nombre específico en un sistema de archivos o sin ir más lejos se pueden usar algoritmos de búsqueda para encontrar soluciones a problemas de optimización, como encontrar el valor máximo de una función, así que son bastante importantes en la programación debido a que seguramente en alguna etapa del desarrollo de una aplicación puede que haya alguno, además es importante que uno comprenda los diferentes algoritmos de búsqueda y cómo utilizarlos, puede mejorar el rendimiento y la eficiencia de sus programas.

Complejidad Temporal

La complejidad es una forma de medir la eficiencia de un algoritmo. Representa el tiempo que tarda un algoritmo en ejecutarse en función del tamaño de la entrada, se mide en O(n). Esto quiere decir que también habrá factores que pueden afectar la eficiencia de un algoritmo como el tamaño de la lista, este es un factor importante a tener en cuenta al elegir un algoritmo de búsqueda debido a que los algoritmos de búsqueda lineal tienen un tiempo de ejecución de O(n), lo que significa

que el tiempo de búsqueda es directamente proporcional al tamaño de la lista. Esto significa que, si la lista tiene el doble de elementos, el algoritmo tardará el doble de tiempo en encontrar el elemento deseado, en cambio los algoritmos de búsqueda binaria tienen un tiempo de ejecución de O(log n), lo que significa que el tiempo de búsqueda aumenta logarítmicamente con el

tamaño de la lista, lo que significa que, si la lista tiene el doble de elementos, el algoritmo tardará aproximadamente el mismo tiempo en encontrar el elemento deseado.



Ordenamiento

El ordenamiento es un algoritmo que organiza los datos de acuerdo a un criterio, como de menor a mayor o alfabéticamente. Así como en la

búsqueda, también hay distintos tipos de ordenamiento:

- Ordenamiento por burbuja (Bubble Sort): Es un algoritmo de ordenamiento simple y fácil de implementar. Funciona comparando cada elemento de la lista con el siguiente elemento y luego intercambiando los elementos si están en el orden incorrecto.

- Ordenamiento por selección (Selection Sort): Este algoritmo funciona encontrando el elemento más pequeño de la lista y lo coloca al inicio, repitiendo el proceso hasta que la lista este completamente ordenada.

• Ordenamiento por Inserción (Insertion Sort): El algoritmo lo que hace es crear una nueva lista ordenada elemento por elemento a partir de la lista existente, insertando cada nuevo elemento en la posición correcta.

• Ordenamiento rápido (Quick Sort): Este algoritmo funciona seleccionando un "pivote" y organiza los elementos menores a un lado y los mayores al otro, esto lo hace ordenando cada parte de manera recursiva llamándose a sí mismo, es mucho más rápido que el Bubble Sort en la mayoría de los casos.

• Ordenamiento por mezcla: Es un algoritmo de ordenamiento eficiente que funciona dividiendo la lista en dos partes, ordenando cada parte y luego fusionando las dos partes ordenadas.

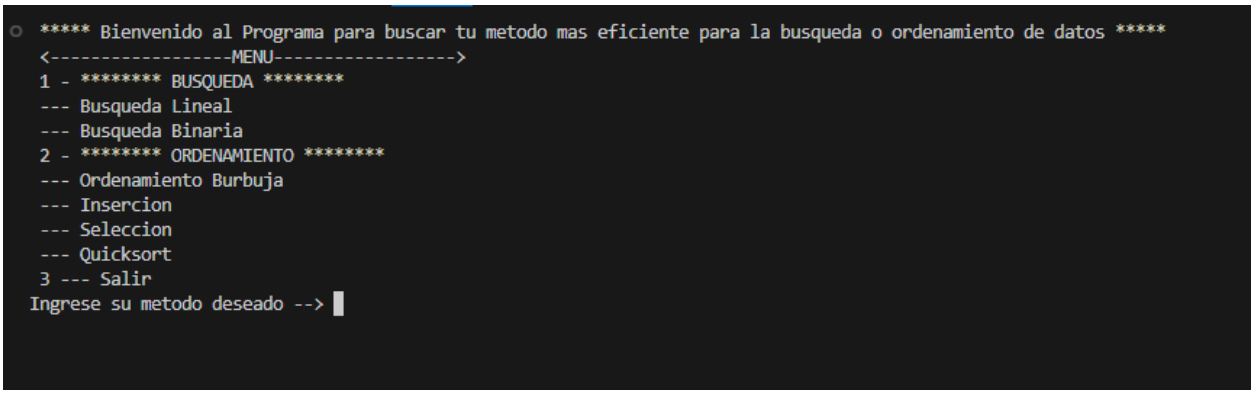
Importancia del ordenamiento

Los algoritmos de ordenamiento son importantes porque permiten organizar y estructurar datos de manera eficiente. Al ordenar los datos, se pueden realizar búsquedas, análisis y otras operaciones de manera más rápida y sencilla.

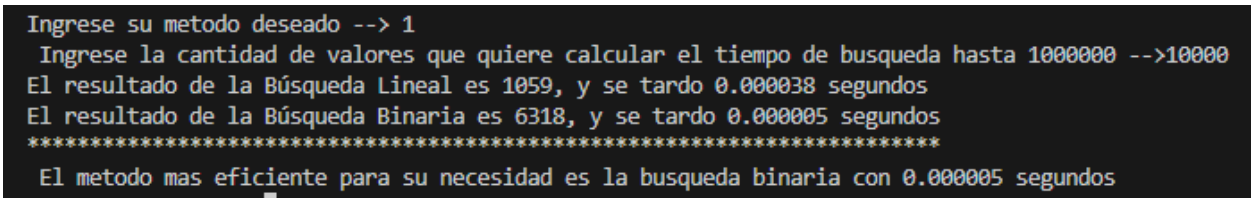
3 – Caso practico

Nosotros simulamos un algoritmo que le permite ver al usuario cual método es más eficiente para su necesidad, el usuario puede elegir si quiere ver cual método de búsqueda le conviene o cual método de ordenamiento

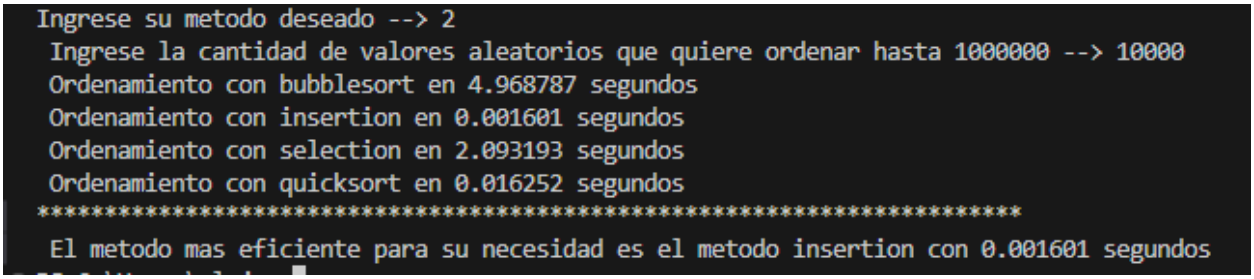
Menu del programa



Opción 1



Opción 2



4 – Metodología Utilizada

1- Se investigaron y revisaron algoritmos típicos de búsqueda y ordenamiento

2- Se desarrollo del menú principal para elegir entre búsqueda u ordenamiento.
3- Implementamos los algoritmos.
4- Se realizo la medición de rendimiento con perf_counter() para obtener tiempos precisos.
5- Se Validaron los resultados y se compararon.

Herramientas utilizadas

- Lenguaje: python 3
- IDE : Visual Studio Code
- Librerías : random, time, os
- Control de versiones: git

5 – Resultados obtenidos

• Se probaron listas de diferentes tamaños: 1,000 - 10,000 - 100,000 elementos.

• Se comprobó cual era el más eficiente en cada caso

6 – Conclusiones

En resumen, los algoritmos de búsqueda y ordenamiento constituyen pilares fundamentales en el desarrollo de software, proporcionando soluciones eficientes, escalables y precisas para la gestión de información, contribuyendo así a la creación de programas más rápidos, organizados y confiables.

7 – Bibliografía

- Aula virtual UTN