Algoritmos Fundamentales: Búsqueda y Ordenamiento

Alumnos:

- Tobias Leiva - tobiasleivaa@gmail.com
- Genaro Luna - genaroluna2808@gmail.com

Materia: Programación I

Profesor/a: Julieta Trapé

Fecha de Entrega: 09/06/2025

Índice

1.Introducción

El trabajo integrador se centra en dos conceptos en la programación: los algoritmos de búsqueda y los métodos de ordenamiento. La elección de estos temas surge del interés por comprender cómo se organizan y manipulan los datos de manera eficiente.

En programación, la capacidad de buscar y ordenar información de manera efectiva es esencial para optimizar el rendimiento de los programas. Permitiendo localizar elementos en listas ordenadas con gran rapidez, siendo muy utilizados en sistemas de archivos y bases de datos.

El objetivo de este trabajo es implementar en Python los algoritmos mencionados, analizar su funcionamiento y comprender su utilidad en la resolución de problemas. Se busca fortalecer los conocimientos adquiridos en la asignatura y desarrollar habilidades mediante la práctica e investigación del tema.

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 una tarea común en muchas aplicaciones, como bases de datos, sistemas de archivos y algoritmos de inteligencia artificial.

Existen diferentes tipos de algoritmos de búsqueda, cada uno eficiente e ideal en diferentes contextos:

- **Búsqueda lineal:** Es el algoritmo de búsqueda más simple, que recorre cada elemento del conjunto de datos de forma secuencial hasta encontrar el elemento deseado. Es fácil de implementar, pero puede ser lento para conjuntos de datos grandes
- **Búsqueda binaria:** Este algoritmo está orientado a listas de datos más grandes, donde los conjuntos de datos deben estar ordenados. Divide 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:** Es un algoritmo de búsqueda que mejora la búsqueda binaria al estimar la posición del elemento deseado en función de su valor. Puede ser más eficiente que la búsqueda binaria para conjuntos de datos grandes con una distribución uniforme de valores.
- **Búsqueda de 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.

La búsqueda es importante en programación porque se utiliza en una amplia variedad de aplicaciones. Algunos ejemplos de uso de la búsqueda en programación son:

- **Búsqueda de palabras clave en un documento o archivo**
- **Búsqueda de archivos en una carpeta o sistema de archivos**
- **Búsqueda de soluciones a problemas de optimización**

- **Búsqueda de la ruta más corta en un gráfico**
- **Filtrar chats o mails por medio de una búsqueda de una palabra clave**

Ordenamiento: El ordenamiento organiza los datos de acuerdo a un criterio, como de menor a mayor o alfabéticamente.

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.

Como en los algoritmos de búsqueda, los algoritmos de ordenamiento son variados y cada uno posee sus ventajas y desventajas:

- **Ordenamiento por burbuja:** 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:** Es otro algoritmo de ordenamiento simple que funciona encontrando el elemento más pequeño de la lista y luego intercambiándolo con el primer elemento. Este proceso se repite hasta que todos los elementos de la lista están ordenados.
- **Ordenamiento por inserción:** Es un algoritmo de ordenamiento que funciona insertando cada elemento de la lista en su posición correcta en la lista ordenada.
- **Ordenamiento rápido o quicksort:** Es un algoritmo de ordenamiento eficiente que funciona dividiendo la lista en dos partes y luego ordenando cada parte de forma recursiva.
- **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.

La importancia de ordenar datos radica en varios factores en un sistema:

- ➔ **Mejora la eficiencia y ejecución de programas con gran volumen de datos**
- ➔ **Permite trabajar con datos de una forma más clara y estructurada**
- ➔ **Hace que el programa sea mucho más escalable y adaptable a cambios**

- ➔ **Garantiza la precisión en la recuperación de datos de forma coherente evitando ambigüedades**
- ➔ **Vuelve mucho más versátil el algoritmo, siendo aplicado en motores de búsqueda, base de datos, servidores web y sistemas de archivos.**

3.Caso Práctico

Como caso práctico, se investigó sobre manejo, lectura y escritura de archivos en Python. Creamos un archivo llamado “Empleados 2024.txt” el cual contenga nombre, apellido y dni de personas que trabajen en una empresa y en python creamos un programa donde lea los datos guardados en el archivo .txt para que los almacene en un diccionario, luego los actualice ingresando nuevos empleados con nombre, apellido y dni para luego ordenarlos base al dni de cada empleado y devolver un archivo “Empleados 2025.txt” con la información actualizada y ordenada.

4.Metodología Utilizada:

- Se investigaron los conceptos de búsqueda y ordenamiento.
- Se desarrolló un programa funcional en Python.
- Se utilizó funciones creadas por los integrantes
- Se aplicó lo aprendido e investigado en el caso práctico.
- Se trabajó en equipo, dividiendo tareas y ayudando al otro compartiendo información e ideas.

5.Resultados Obtenidos:

- Resolución del ejercicio planteado
- Se pudo utilizar un algoritmo de búsqueda y uno de ordenamiento en el mismo caso práctico
- Se logró ejemplificar la función e importancia de estos
- Se manejo una base de datos pequeña sin complicaciones con estos algoritmos
- Comprensión del cómo funciona cada uno
- Modificación del algoritmo de búsqueda para adaptarlo al caso práctico
- Investigación sobre lenguaje Python
- Aprendizaje sobre nuevas funciones, utilidades y una forma de cargar leer y escribir documentos
- Los integrantes del grupo obtuvieron una mayor comprensión del lenguaje Python

6.Conclusión:

A lo largo de este trabajo integrador, aprendimos a cómo funcionan los algoritmos de búsqueda y ordenamiento. Comprendimos la lógica detrás de distintos algoritmos, su eficiencia y cuándo conviene usar cada uno.

Una dificultad que tuvimos fue comprender al principio cómo funcionan algunos algoritmos más complejos. Lo resolvimos investigando, viendo ejemplos y ayudándonos entre todos para entender mejor los conceptos.

Estos temas tienen una gran utilidad en programación, ya que permiten organizar y acceder a la información de manera más rápida. Además, son la base para muchos sistemas reales, bases de datos y aplicaciones que manejan grandes volúmenes de datos.

Una mejora que consideramos importante para el futuro es la organización del equipo. Si bien logramos completar todas las tareas, al principio hubo dificultades para coordinar los tiempos y mantener una comunicación constante. Esto nos enseñó la importancia de planificar mejor desde el inicio y definir claramente las responsabilidades de cada integrante.

7.Bibliografía:

Algoritmos de busqueda y ordenamiento:

https://medium.com/@vyankateshpareek733/sorting-algorithms-11a9057d688f

Video, Ordenamiento por selección (2021), Canal de Chio Code:

Ordenamiento por Selección | Selection Sort

Video, Ordenamiento Burbuja | Bubble Sort

https://www.youtube.com/watch?v=pqZ04TT15PQ

Luigi Box, TIpos de Algoritmos de Búsqueda:

https://www.luigisbox.es/glosario-de-busqueda/algoritmo-de-busqueda/

GitHub de gaudino, Metodos de ordenamiento:

https://github.com/gbaudino/MetodosDeOrdenamiento

8. Anexos

Link del video: https://www.youtube.com/watch?v=rR7j9ZyaFIY