“Búsqueda lineal”

Sistema de Tienta Electrónica TechStore

Este taller tiene como objetivo demostrar e implementar el algoritmo de **“Búsqueda lineal”** a través de un contexto practico, donde se aplican conocimientos en programación y complejidad de algoritmos.

En este caso, cuando se habla de **“Búsqueda lineal”,** se habla de un algoritmo de complejidad espacial O(n) y una complejidad de temporal O(1), es decir, en el peor caso tiene una complejidad de O(n) y en el mejor caso O(1) .

La **“Búsqueda lineal”** no requiere que los datos estén ordenados, es un algoritmo el cual es estable y rápido SOLO para ser aplicado con pocos datos, debido a que a medida que aumenta el número de datos, el algoritmo debe hacer el mismo número o más de procesos lo que a gran escala lo vuelve inutilizable.

Actividad 1: Análisis de Complejidad

¿Cuál es la complejidad temporal de cada función implementada?

Cada una de las funciones implementadas, debe recorrer al menos 1 vez la lista o el arreglo, eso quiere decir que tienen una complejidad temporal O(n)

¿En qué casos la búsqueda lineal es eficiente?

La búsqueda lineal es eficiente cuando el dato buscado está en la primera posición del arreglo o lista y cuando la cantidad de datos es pequeña.

¿Cuándo sería mejor usar otro algoritmo de búsqueda?

Por lo general cuando se debe manejar un gran número de datos, ya que la búsqueda lineal se vuelve lenta, por ende es mejor usar búsqueda binaria O(log n) u otro tipo conocido como tablas bash en casos específicos.

¿Qué pasa si la lista está vacía?

Si la lista esta vaciá, el bucle **for** no se ejecuta, por ende devuelve **None** o una lista/arreglo vacío, esto indica que no se encontró ningún elemento, por ende se devuelve o imprime un mensaje al usuario, estilo **messagebox.showinfo** o un simple **print** (en el caso de que sea por consola)

¿Cómo manejar búsquedas con mayúsculas/minúsculas?

Las mayúsculas o minúsculas se manejan desde código, con el comando **.lower()** que va después de la variable en la que se almacenan los datos ingresados por el usuario, de esta manera **variable.lower** , eso permite que sin importar como escriba el usuario la entrada se pueda procesar

¿Cómo buscar texto parcial en nombres?

En este caso las funciones comparan cadenas completas con **= =,** en el caso de que se quiera hacer con entradas parciales, se debe usar un **in**, de manera que en el código la entrada se convierta y se relacione con el dato más similar, ejemplo:

```
if nombre.lower() in producto['nombre'].lower():

return producto
```

Conclusion

La búsqueda lineal es un algoritmo sencillo pero muy útil para entender cómo funcionan los recorridos dentro de una lista o arreglo. A pesar de no ser el método más rápido, cumple perfectamente cuando se trabaja con pocos datos o estructuras pequeñas, como en este caso con productos y empleados. Su ventaja principal es que no necesita que los datos estén ordenados, y es muy fácil de implementar y entender.

Durante el desarrollo del sistema TechStore se aplicó la búsqueda lineal en distintos contextos, demostrando que con una buena estructura de código se pueden reutilizar las mismas funciones en diferentes entornos, tanto en consola como en una interfaz gráfica. En general, este trabajo permitió reforzar la lógica de programación, el manejo de errores y la comprensión de la eficiencia de los algoritmos en situaciones reales.