OscarRojo

Nivel: 1
Tipo: individual
Modo: obligatorio
Dificultad: intermedio

Skootik Mobile Technologies

https://skootik.com
info@skootik.com

Comprender que para llegar a un mismo fin existen diferentes caminos. Estos caminos en el mundo de la computación son los algoritmos, cada uno con sus virtudes y sus defectos.

Algoritmos de búsqueda

Existen algoritmos que permiten buscar un elemento dentro de una colección. Estas operaciones son muy empleadas en computación para la búsqueda de una determinada información de ahí su gran importancia. Es el recopilatorio de un conjunto de instrucciones que están diseñadas para localizar un elemento con ciertas propiedades dentro de una estructura de datos y estas suelen ser bases de datos, arrays, listas, ...

Para entender mejor de qué estamos hablando lo mejor es poner unos ejemplos : la búsqueda de una persona para el acceso a un evento en un sistema de registros compuesto por un array, la ubicación de una materia prima en un proceso de fabricación compuesta por miles de diferentes referencias todas ellas guardadas en una base de datos, la cantidad en euros de la factura de la luz de un mes y año dados y guardados en un excel, ...

Si el dato está registrado y el contenedor de datos mínimamente estructurado se podrá acceder a esta información mediante diferentes algoritmos de búsqueda. En este caso nos centraremos en dos:

1.- Búsqueda secuencial
2.- Búsqueda binaria

1.- Búsqueda secuencial: A partir del argumento de búsqueda, este algoritmo compara uno a uno los elementos del arreglo o base de datos hasta dar con el elemento en caso de que exista.
Su ventaja es que el vector no tiene que estar ordenado necesariamente, sin embargo, es lento ya que si el conjunto de datos tiene muchos elementos lo recorre por completo.

2.- Búsqueda binaria: Este algoritmo permite buscar de una manera más eficiente un dato dentro de un arreglo o base de datos. Se determina el elemento central del conjunto de datos y se compara con el valor que se esta buscando, en caso de que coincida se termina la búsqueda y en caso de no ser así se determina si el dato que se pretende buscar es mayor o menor que el elemento central, de esta forma se elimina una mitad del arreglo junto con el elemento central. Este proceso se repetirá hasta encontrarlo o tener un único elemento en el conjunto de datos.
Para poder aplicar este algoritmo se requiere que el conjunto de datos esté ordenado. Es más rápido que el secuencial ya que utiliza la táctica del divide y vencerás.

Disfruta de este ilustrativo GIF animado para entender mejor como funcionan ambos algoritmos:
https://k60.kn3.net/4/B/3/6/C/E/05E.gif

EJERCICIO A RESOLVER: Tenemos la siguiente lista de elementos: [3, 56, 21, 33, 874, 123, 66, 1000, 23, 45, 65, 56].
1.- Construye tu propio algoritmo para ordenarlo de menor a mayor.
2.-

Busca el número 875 utilizando el algoritmo secuencial y el binario. En cada iteración se debe sumar +1 de modo que al final del programa se debe indicar el número de iteraciones realizadas por cada algoritmo hasta encontrar el elemento.
3.- Realiza el análisis en Notación Big O (visto en la tarea #44) y describe tu conclusiones en un documento de texto.

Debes subir a tu repositorio GitHub tanto el programa (en el lenguaje de programación que hayas elegido) y el documento de texto explicativo y razonado sobre el rendimiento y los tiempos de ejecución de cada algoritmo en notación O Grande. Por supuesto, no te olvides del diccionario.

#HASHTAGS *(etiquetas de ayuda para búsqueda de información relevante)*

#algoritmo #base-de-datos #búsqueda-secuencial #búsqueda-binaria #notación-Big-O

LINKS DE INTERÉS

https://www.youtube.com/watch?v=HmUpRHn31FU
https://k60.kn3.net/4/B/3/6/C/E/05E.gif
https://es.wikipedia.org/wiki/Algoritmo_de_b%C3%BAsqueda

DICCIONARIO

búsqueda-secuencial | búsqueda-binaria | notación-Big-O

PUNTUACIÓN

Programación: 3
Redes: 1
Seguridad: 1
Algoritmia: 6