Universidad Autónoma deBaja California

Facultad de Ingeniería, Arquitectura yDiseño

Ingeniero en Computación

Asignatura:

Programación Estructurada

Actividad 9:

FUNCIONES Y METODOS DE ORDENACION Y BUSQUEDA

Brayan Arturo Rocha Meneses

Matricula:

371049

Ensenada Baja California 10 de Octubre del 2023

Introducción:

En este tema se prioriza sobre la búsqueda y como ordenar los vectores de una manera correcta, además nos centramos en crear nuestra propia librería.

Estas herramientas permiten a los programadores organizar y buscar datos de manera eficiente en una variedad de aplicaciones. La ordenación se refiere a la tarea de reorganizar elementos en una colección en un orden específico, como ascendente o descendente, mientras que la búsqueda se trata de encontrar un elemento particular en una colección de datos.

Competencia:

Utilizar la recursión, aplicando la abstracción procedimental en la estrategia de solución de problemas complejos, para ofrecer una perspectiva basada en sus propias definiciones al plantear una solución, con actitud analítica y organizada.

Fundamentos:

Ordenación Ascendente y Descendente:

Las funciones de ordenación en C++ permiten organizar elementos en un contenedor, como un arreglo o un vector, ya sea en orden ascendente o descendente. Algunos algoritmos de ordenación comunes en C++ son std::sort() para ordenar de manera ascendente y std::sort() con un comparador personalizado para ordenar de manera descendente.

Búsqueda en Vectores:

Para buscar un elemento en un vector en C++, puedes usar métodos como la búsqueda secuencial o la búsqueda binaria si el vector está ordenado. La búsqueda secuencial implica recorrer el vector elemento por elemento, mientras que la búsqueda binaria es eficiente para vectores ordenados.

Búsqueda en Matrices:

La búsqueda en matrices implica buscar un elemento específico en una estructura bidimensional. Puedes usar bucles anidados para recorrer la matriz y buscar el elemento deseado.

Ordenación de Vectores:

La ordenación de vectores se realiza utilizando la función std::sort() de la biblioteca estándar de C++. Puedes especificar un comparador personalizado si deseas un ordenamiento personalizado. Aquí tienes un ejemplo de ordenación de un vector de enteros:

Ordenación de Matrices:

La ordenación de matrices implica definir un criterio de ordenación específico para las filas o columnas. Puedes aplicar algoritmos de ordenación a las filas o columnas de la matriz según tus necesidades.

Algoritmos de Ordenación:

Ordenación Burbuja (Bubble Sort):

El algoritmo de ordenación burbuja es simple pero no muy eficiente para grandes conjuntos de datos. Compara pares de elementos adyacentes y los intercambia si están en el orden incorrecto. Este proceso se repite hasta que no se requieran más intercambios**.**

Ordenación por Selección (Selection Sort):

En el algoritmo de ordenación por selección, se busca el elemento más pequeño en el vector y se coloca al principio. Luego, se busca el segundo elemento más pequeño y se coloca en la segunda posición, y así sucesivamente. Es adecuado para pequeños conjuntos de datos.

Ordenación por Inserción (Insertion Sort):

El algoritmo de ordenación por inserción recorre el vector de izquierda a derecha, colocando cada elemento en su posición correcta. Es eficiente para conjuntos de datos pequeños y casi ordenados.

Es importante seleccionar el algoritmo de ordenación y búsqueda adecuado según la naturaleza de los datos y los requisitos de rendimiento de tu aplicación. Cada algoritmo tiene sus ventajas y desventajas, por lo que es crucial elegir el que mejor se adapte a tus necesidades específicas.

Procedimiento:

ACTIVIDAD 9

Realiza programa en C utilizando librería propia, el programa deberá tener el siguiente menú.

MENÚ
1.- LLENAR VECTOR
2.- LLENAR MATRIZ
3.- IMPRIMIR VECTOR
4.- IMPRIMIR MATRIZ
5.- ORDENAR VECTOR
6.- BUSCAR VALOR EN VECTOR
0.- SALIR

NOTA: El programa deberá repetirse cuantas veces lo desee el usuario, Validado el menú con la función vali_num

I N S T R U C C I O N E S

1.- LLENAR VECTOR .- Llenar vector con 15 números, los números generados aleatoriamente, los números entre el rango de 100 al 200 **(no repetidos)**
2.- LLENAR MATRIZ .- Llenar la matriz de 4x4 con con números generados aleatoriamente, números entre el rango de 1 al 16 **(no repetidos)**
3.- IMPRIMIR VECTOR .- Imprime el vector que se envíe, donde la función recibe como parámetro el vector,tamaño, nombre del vector.
4.- IMPRIMIR MATRIZ.- Imprime la matriz sin importar el tamaño de la matriz recibiendo como parámetros la matriz, la cantidad de renglones y columnas, así como nombre que se le dará a la matriz
5.- ORDENAR VECTOR.- Usar función que ordene el vector por el método de ordenación de la Burbuja mejorada.
6.- BUSCAR VALOR EN VECTOR.- Buscar un valor en el vector usando el método de búsqueda secuencial.
0.- SALIR

Conclusiones:

Las funciones de ordenación permiten organizar elementos en un orden específico, ya sea ascendente o descendente, y pueden adaptarse a necesidades específicas mediante comparadores personalizados. Por otro lado, los métodos de búsqueda son cruciales para encontrar elementos específicos en conjuntos de datos, ya sea a través de una búsqueda secuencial, búsqueda binaria o implementaciones personalizadas.

En cuanto a los algoritmos de ordenación, se destacaron el algoritmo de burbuja, el algoritmo de selección y el algoritmo de inserción. Cada uno de estos algoritmos tiene sus propias ventajas y desventajas, y la elección del algoritmo adecuado depende de la cantidad de datos, la estructura de datos y los requisitos de rendimiento de la aplicación.

Anexo:

https://github.com/BrayanArM98/Estructurada/blob/main/ACTIVIDAD%209%20ESTRUCTURADA.pdf