INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

Programación de estructuras de datos y algoritmos fundamentales

Evidencia 1

Alumno:

Jesús Ricardo Guerrero Silvestre A00835912

Profesor:

David Alonso Cantú Delgado

Fecha de entrega:

10 de septiembre del 2023

Reflexión

Durante este primer semestre trabajé en las bases que me ayudarán a poder trabajar con estructuras de datos y algoritmos fundamentales, como lo dice el nombre del curso. Trabajé con distintos tipos de algoritmos, los de ordenamiento y los de búsqueda. Ambos me parecieron interesantes y al igual que sus aplicaciones considero que son importantes poder comprender los procesos de distintas herramientas digitales con las que interactuamos en nuestro día a día.

Algoritmos de ordenamiento:

1) Swap Sort: El método de intercambio se basa en comparar los elementos del arreglo e intercambiarlos si su posición no es la desea. Por ejemplo, si se busca acomodar una lista de enteros del menor al mayor primero se buscará el número más chico de la lista para acomodarlo al inicio y así ir avanzando hasta acomodar todos.

   Complejidad: O(n^2)
2) Bubble Sort: El algoritmo atraviesa una lista y compara valores adyacentes, intercambiándolos si no están en el orden correcto. Es un algoritmo lento, pero con una baja complejidad lo que permite entenderlo fácilmente.

   Complejidad: O(n^2)
3) Selection Sort: Este algoritmo recibe su nombre de la forma en que itera a través del arreglo: Selecciona el elemento más pequeño actual y lo cambia de lugar.

   Complejidad: O(n^2)
4) Insertion Sort: Se selecciona el primer elemento y después se va verificando si el próximo elemento es menor o mayor, si es menor se coloca antes del elementos seleccionado hasta que esté acomodado con todos los elementos en caso de que existan más.

   Complejidad: O(n^2)
5) Merge Sort: Se hace uso de la recursividad. Se divide la lista en 2 partes para después acomodar ambas listas del menor al mayor.

   Complejidad: O($n\ log\ (n)$)

6) Quick Sort: el mismo funciona seleccionando un elemento como pivote y dividiendo la matriz dada alrededor del pivote elegido.
   Complejidad: O($n\ log\ (n)$)
7) Shell: es una mejora del Método de Ordenamiento por Inserción. Compara elementos separados por un espacio de varias posiciones, esto permite que un elemento haga "pasos más grandes" hacia su posición esperada, el mismo finaliza con un Ordenamiento por inserción simple.
   Complejidad: O(n^2)

Algoritmos de búsqueda:

1) Búsqueda secuencial: Consiste en ir comparando el elemento que se busca con cada elemento del arreglo hasta cuando se encuentra.
   Complejidad: O(n)
2) Búsqueda binaria: La Búsqueda Binaria, compara si el valor buscado está en la mitad superior o inferior. En la que está, subdivido nuevamente, y así sucesivamente hasta encontrar el valor.
   Complejidad: O($log_2(n)$)