INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES

DE MONTERREY

TC1031 - Programación de estructuras de datos y algoritmos fundamentales

Profesor: David Sánchez

Actividad 3 - Implementación de una Linked List

Grupo 604

Diana María Arámburo Lozano | A01646337

Camila Gomez Godinez | A01639319

Gabriela Ruelas Gaytán | A01640880

Emilio Guzmán Flores | A01643485

Samantha Mailen Gallardo Mota | A01640886

Martes 16 de septiembre del 2025

Actividad 3 - Implementación de una Linked List

Explicación breve de la Linked List

La Linked List es una estructura de datos lineal formada por nodos que contienen un valor y un puntero hacia el siguiente nodo. Una Linked List, a diferencia de los arreglos, no utiliza posiciones contiguas en la memoria; cada nodo apunta al siguiente lugar. Esto permite que insertar o eliminar elementos en posiciones conocidas sea eficaz, dado que no se requiere desplazar los otros elementos.

Los componentes principales son los siguientes:

- Head: puntero hacia el primer nodo de la lista (nulo si está vacía).
- Nodo: es una estructura interna compuesta por dos campos, que son el valor (data) y el puntero al siguiente nodo (next).
- Size: es un contador de elementos que resulta útil para hacer consultas rápidas y validaciones.

Justificación de la elección de Linked List

- Eliminaciones e inserciones dinámicas: a diferencia de un arreglo, no es necesario mover elementos ni establecer una dimensión fija desde el principio.
- Uso eficaz de la memoria: la lista se amplía o disminuye en función de las operaciones utilizando únicamente la memoria requerida para los nodos.
- Flexibilidad en las operaciones: es fácil agregar elementos al principio, al final o en cualquier otra posición determinada.

Básicamente, el beneficio que se obtiene al realizar operaciones de modificación hace que sea apropiado en situaciones dinámicas, a pesar de que el costo de acceder a un elemento intermedio es más alto que en un arreglo por que no existe acceso directo por índice.

Complejidad de las operaciones principales

Sea n el número de nodos de la lista:

- insertAtBeginning(value)

Inserta un nodo al inicio, actualizando el puntero head.

Complejidad: O(1).

- insertAtEnd(value)

Recorre la lista hasta el último nodo y agrega al final.

Complejidad: O(n), ya que puede ser necesario visitar todos los nodos.

- insertAtPosition(pos, value)

Recorre hasta la posición deseada y enlaza el nuevo nodo.

Complejidad: O(n) en el caso general.

- deleteValue(value)

Recorre la lista hasta encontrar la primera coincidencia, ajusta punteros y libera memoria.

Complejidad: O(n).

- search(value)

Compara cada nodo hasta encontrar coincidencia o llegar al final.

Complejidad: O(n).

- display()

Recorre todos los nodos imprimiendo su contenido.

Complejidad: O(n)