**UNIDAD 3**

**ALGORISMOS # 2**

**DANCY DAYANA GOMEZ GALINDEZ**

**ANA SOFIA ANGEL GALAN**

**Código: 1060296429**

****

**Docente: JORGE GAMARRA**

**UNIVERSIDAD REMINGTON DE CALI**

**Facultad de Ingeniería**

**Ingeniería En Sistema**

**Cali, agosto 2025**

**Parte 3 – Análisis**

1. **Identifica cuáles operaciones son O(1) y cuáles son O(n).**

**O(1)**:

* Insertar Inicio: Solo modifica referencias sin recorrer la lista.
* Eliminar Inicio: Elimina el primer nodo cambiando la cabeza.

**O(n)**:

* Insertar Final: Recorre toda la lista para encontrar el último nodo.
* Insertar En Posición: Recorre hasta la posición deseada.
* Eliminar Final: Recorre hasta el penúltimo nodo.
* Eliminar En Posición: Recorre hasta la posición deseada.

Las operaciones O(1) no dependen del tamaño de la lista, ya que no requieren recorrerla. En cambio, las operaciones O(n) sí lo hacen, ya que deben iterar por los nodos para encontrar el punto de inserción o eliminación.

1. **Explica por qué cada operación tiene esa complejidad.**

**T**eniendo en cuenta el ejercicio lo que se puede observar es que O(1) es una lista enlazada el primer nodo está directamente accesible a través de la referencia head. No se necesita recorrer la lista.

O(n) – Búsqueda de un valor

Por qué: Para encontrar un valor, se debe recorrer nodo por nodo desde el inicio hasta encontrarlo (o llegar al final).

Ejemplo: Buscar un valor inexistente en una lista de 100.000 nodos requiere revisar todos los nodos.

1. **Compara los tiempos medidos y confirma si los resultados coinciden con la teoría.**

-En lista ordenadas o invertidas los tiempos fueron mayores (peor caso de Quicksort).

**Invertidos y ordenados** → tiempos altos (peor caso).

**Aleatorios** → tiempos bajos (mejor promedio).

**Medio** → tiempos intermedios.

**Preguntas de reflexión**

1. **¿Qué ventajas y desventajas tiene una lista simplemente enlazada frente a un arreglo tradicional en términos de rendimiento y uso de memoria?**

Ventajas de la lista enlazada

* Inserciones/eliminaciones rápidas al inicio.
* No requiere redimensionar ni mover elementos.
* Ideal para estructuras dinámicas donde el tamaño cambia frecuentemente.

Desventajas de la lista enlazada

* Acceso lento por índice.
* Mayor uso de memoria por punteros.
* Menor eficiencia en caché por no ser contigua.

¿Cuándo usar cada una?

**Lista enlazada:** cuando necesitas muchas inserciones/eliminaciones en posiciones variables.

**Arreglo:** cuando necesitas acceso rápido por índice y el tamaño es relativamente estable.

1. **Si se quisiera implementar la eliminación de un nodo específico por valor y no por índice, ¿cómo afectaría eso la complejidad temporal?**

Eliminar un nodo por **valor** en una **lista simplemente enlazada** implica buscar el nodo que contiene ese valor y luego ajustar los punteros para eliminarlo Eliminación por valor: ¿Qué implica? recorrer la lista desde el inicio hasta encontrar el nodo con el valor deseado.

**Ajuste de punteros**: una vez encontrado, se actualiza el puntero del nodo anterior para que salte el nodo a eliminar. Si el valor está cerca del inicio, el tiempo será menor. Pero en el peor caso (valor al final o no existe), se recorre toda la lista.

1. **¿Cómo cambiaría la complejidad temporal si en lugar de agregar al final de la lista se insertara un nodo en una posición aleatoria en cada operación?**

En una lista simplemente enlazada, insertar en una posición aleatoria implica dos pasos:

1. Recorrer hasta la posición deseada (porque no hay acceso directo por índice).
2. Insertar el nuevo nodo ajustando punteros.