



Universidad Tecnológica  
del Norte de Guanajuato

Organismo Público Descentralizado del Gobierno del Estado

**"Educación y progreso para la vida"**

UNIVERSIDAD TECNOLÓGICA DEL NORTE DE GUANAJUATO

Licenciatura en Ingeniería en Tecnologías de la Información e Innovación

Digital -Especialidad Desarrollo de Software

Multiplataforma

Estructura de datos

**Actividad Listas en java en Visualgo.net**

Unidad 2

Fernando Miguel Olvera Juárez

No. De Control 1224100597

Profesor Barron

Dolores Hidalgo C.I.N. a martes 9 de octubre del 2024.

GTID0141

## PREGUNTAS

### 1. Qué pasa con los punteros al insertar o eliminar un nodo

Al insertar o eliminar un nodo, se actualizan los punteros del nodo anterior y del siguiente para mantener la lista conectada correctamente.

### 2. Cómo afecta la posición del nodo al tiempo de búsqueda

Buscar un nodo depende de su posición: al inicio es rápido, en medio tarda un poco y al final toma más tiempo porque hay que recorrer toda la lista.

### 3. Ventajas de recorrer una lista enlazada frente a un arreglo

Las listas enlazadas permiten insertar y eliminar nodos fácilmente sin mover otros elementos, además no necesitan un tamaño fijo como los arreglos.

### 4. Cómo comprobar si una lista está vacía en Java

Para verificar si la lista está vacía, se comprueba si la cabeza apunta a null:

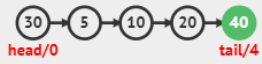
```
if (head == null) {  
    // lista vacía  
}
```

## EVIDENCIAS

The screenshot displays the VISUALGO website interface. At the top, navigation tabs include 'LINKED LIST', 'STACK', 'QUEUE', 'DLL', and 'DEQUE'. The 'LINKED LIST' tab is active. The main area shows a linked list with four nodes containing values 30, 5, 10, and 20. The 'head' pointer is at index 0, 'tmp' is at index 2, and 'tail' is at index 3. A search operation is being performed for the value 10. A pink box indicates: 'Found value v = 10 at this highlighted vertex so we return index 2. The whole operation is O(n)'. Below this, a yellow box contains the following Java code snippet:

```
if empty, return NOT_FOUND  
index = 0, tmp = head  
while (tmp.item != v)  
    index++, tmp = tmp.next  
if tmp == null  
    return NOT_FOUND  
return index
```

At the bottom, there is a control panel with a menu on the left containing 'Create(A)', 'Search', 'Insert', and 'Remove'. The 'Search' option is selected. To the right of the menu, there are input fields for 'i = 0 (Head), specify v =' and 'i = N (After Tail), specify v ='. The value '40' is entered in the second field. A 'Go' button is next to the input fields. On the far right, a box says 'specify both i in [1..N-1] and v ='.



### Insert 40 at tail

tail points to vtx.

The whole operation is  $O(1)$  if we maintain the tail pointer.

```
Vertex vtx = new Vertex(v)
```

```
tail.next = vtx
```

```
tail = vtx
```