SESIÓN **/06**

LISTAS ENLAZADAS SIMPLES

1. Búsqueda
2. Modificación
3. Eliminación
4. Ordenamiento

**INTRODUCCIÓN**

* La clase anterior conocimos las Listas enlazadas simples. Vimos sus principales características, la forma cómo se representan y conocimos los algoritmos de recorrido e inserción para este tipo de lista.
* En la presente sesión revisaremos los algoritmos para realizar otras operaciones sobre una Lista enlazada simple: Búsqueda, modificación, eliminación y ordenamiento.
* Culminaremos comparando la eficiencia de las operaciones en un Vector y en una Lista enlazada simple.

1. **BÚSQUEDA**

Se trata de ubicar un valor dentro de la Lista enlazada. Se recorre la Lista comparando el valor buscado con cada elemento. El algoritmo que se utiliza es búsqueda secuencial.

Algoritmo de búsqueda

1. Se asigna el valor de la variable inicio a una variable p.
2. Repetir los pasos 3 y 4 hasta encontrar el dato buscado ó hasta llegar al final de la Lista.
3. Comparar el dato buscado con el valor almacenado en el nodo.
4. Pasar al siguiente nodo de la Lista.

Si se encuentra el dato buscado, obtenemos la dirección de memoria dónde está almacenado.

1. **MODIFICACIÓN**

Consiste en reemplazar un valor que se encuentra almacenado en la Lista enlazada.

Algoritmo de modificación

1. Buscar en la Lista enlazada el valor a reemplazar (con el algoritmo de búsqueda secuencial).
2. Asignar el valor de reemplazo en la respectiva dirección de memoria.

Primero se debe buscar el valor a reemplazar para conocer la dirección de memoria donde se encuentra almacenado.

1. **ELIMINACIÓN**

Consiste en desenlazar de la Lista a un determinado elemento, liberando además el espacio de memoria que ocupaba.

Antes de realizar esta operación, es necesario realizar las siguientes verificaciones:

* Si la Lista está vacía, se deberá mostrar un mensaje.
* Si la Lista no está vacía, se buscará el valor a eliminar.
* Si el valor a eliminar no está en la Lista, se deberá mostrar un mensaje.

Según la ubicación del nodo a eliminar, se presentan los siguientes casos:

* Eliminación al inicio de la Lista.
* Eliminación entre dos elementos.
* Eliminación al final de la Lista.

**3.1 CASO 1: Eliminación al inicio de la Lista**

1. Hacer que la variable inicio apunte al segundo elemento de la Lista.

2. Liberar el espacio de memoria.

* 1. **CASO 2: Eliminación entre dos elementos**

1. Hacer que la variable actual apunte al nodo a eliminar.
2. Hacer que la variable anterior apunte al nodo que se encuentra antes del nodo a eliminar.
3. Hacer que el campo de enlace del nodo anterior apunte al sucesor del nodo actual.
4. Liberar el espacio de memoria.
   1. **CASO 3: Eliminación al final de la Lista**

Se aplica el algoritmo descrito en el CASO 2.

1. **ORDENAMIENTO**

Se trata de organizar los valores, almacenados en la Lista enlazada, bajo algún criterio. El ordenamiento puede ser ascendente ó descendente. Si no se especifica, se asume un ordenamiento de menor a mayor. El algoritmo que se utiliza es el ordenamiento burbuja.

***EJEMPLO:***

***Ordena los elemento de la siguiente Lista enlazada***

inicio

04

29

03

null

13

Se asume ordenamiento ascendente.

inicio

03

04

13

null

29

***Sólo se intercambia el contenido del campo de datos.***

***El contenido del campo de enlace queda igual.***

**CONCLUSIÓN**

Para modificar o eliminar un elemento de una Lista enlazada, primero se lo debe buscar. Así obtenemos la dirección de memoria dónde se encuentra almacenado. En el algoritmo de ordenamiento sólo se intercambia el contenido del campo de datos. Las operaciones de inserción y eliminación son más eficientes en una Lista enlazada que en un Vector.

**REFERENCIAS BIBLIOGRÁFICAS**

* Cairo, O.; Guardati, S. (2008). Estructuras de datos. 3ra. Edición. México D.F., México: McGraw Hill.
* Instituto NIIT (2011). Data Structures and Algorithms. Student guide.