**LISTAS LIGADAS O ENLAZADAS**

**Fundamentos teóricos**

Una lista es aquella estructura que representa un numero contable de valores ordenados donde un mismo valor puede repetirse y considerarse un valor distinto a otro ya existente. Las listas son consideradas secuencias de valores.

Una lista enlazada es una colección lineal de elementos llamados nodos. El orden entre ellos se establece mediante punteros; direcciones o referencias a otros nodos.

Una lista enlazada, ligada o encadenada es un conjunto de elementos en los que cada elemento contiene la posición o dirección del siguiente elemento de la lista.

Se pueden usar punteros (apuntadores) para implementar una lista enlazada. Los inconvenientes de las litas contiguas se eliminan con las listas enlazadas. Se pueden almacenar los elementos de una lista lineal o posiciones de memoria que no sean contiguas o adyacentes.

Un puntero es una variable cuyo valor es la dirección o posición de otra variable.

Cada elemento de la lista enlazada debe tener al menos dos campos: un campo tiene el valor del elemento y un campo (enlace, link) que contiene la posición del siguiente elemento, es decir, su conexión, enlace o encadenamiento.

Cada nodo contiene al menos un campo “pointer” apuntando a otro nodo.

Es necesario tener un pointer que apunte a la cabeza de la estructura para así saber por dónde comenzar.

########EJEMPLO#######

Un nodo esta constituido por dos partes:

* Un campo INFORMACION que será del tipo de los datos de los elementos que se quiera almacenar en la lista.
* Un campo LIGA de tipo puntero, que se utiliza para establecer la liga o el enlace con otro nodo de la lista.

Los datos se almacenan en forma dinámica en un alista enlazada; se crea cada nodo según sea necesario.

Un nodo puede contener datos de cualquier tipo, incluyendo objetos de otras clases.

Son estructuras de datos lineales no adyacentes.

**Ventajas de las listas ligadas**

Una lista enlazada es apropiada cuando el número de elementos de datos que se van a representar en un momento dado es impredecible.

Las listas enlazadas son dinámicas, por lo que la longitud de una lista puede incrementarse o reducirse, según sea necesario.

Las listas enlazadas se llenan cuando el sistema no tiene suficiente memoria para satisfacer las peticiones de asignación dinámica de almacenamiento.

Para mantener las listas enlazadas en orden, se inserta cada nuevo elemento en el punto apropiado en la lista. Los elementos existentes de una lista no necesitan moverse.

Los nodos de las listas enlazadas no se almacenan contiguamente en memoria. Sin embargo, en sentido lógico los nodos de una lista enlazada parecen estar contiguos.

**\*Solo necesita un valor y el siguiente (apuntador).**

######EJEMPLO#####

**Clasificación de las listas**

* Listas enlazadas simples

Es una lista enlazada de nodos, donde cada nodo tiene un único campo de enlace.

Una variable de referencia contiene al primero nodo, cada nodo (excepto el ultimo) enlaza con el siguiente nodo, y el enlace del ultimo nodo contiene NULL para indicar el final de la lista. Aunque normalmente a la variable de referencia se le suele llamar top, se le podría llamar como desee.

* Listas enlazadas dobles

Un tipo de lista enlazada mas sofisticado es la lista doblemente enlazada o lista enlazada de dos vías.

Cada nodo tiene dos enlaces: uno apunta al nodo anterior, o apunta al valor NULL si es el último nodo.

* Listas circulares simples

En una lista enlazada circular, el primero y el ultimo nodo están unidos juntos. Esto se puede hacer tanto para listas enlazadas simples como para las doblemente enlazadas.

Para recorrer una lista enlazada circular podemos empezar por cualquier nodo y seguir la lista en cualquier dirección hasta que se regrese hasta el nodo original. Desde otro punto de vista, estas listas pueden ser vistas como listas sin comienzo ni fin. Este tipo de listas es el mas usado para dirigir buffers para “ingerir” datos, y para visitar todos los nodos de una lista a partir de uno dado.

* Lista circular doblemente enlazada

Una lista doblemente enlazada en la que el ultimo elemento se enlaza al primer elemento y viceversa. Esta lista se puede recorrer de modo circular (en anillo) tanto en dirección directa (<<adelante>>) como inversa (<<atrás>>).

**Operaciones básicas**

* **Insertar**
* **Obtener**
* **Eliminar**
* **Está vacío (comprobación)**

Una lista enlazada se compone de una serie de nodos enlazados mediante punteros.

Cada nodo es una combinación de dos partes: un tipo de dato (entero, real, doble, carácter, etc.) y un enlace (puntero) al siguiente nodo.

En C++ se puede definir un nodo mediante un nuevo tipo de dato con las palabras reservadas struct o class que contienen las dos partes citadas.

**El operador -> de selección de un miembro**

Si p es un puntero a una clase (o una estructura) y m es un miembro de la clase, entonces **p -> m** accede al miembro m de la clase o estructura apuntado por p.

El símbolo (**->**) se considera como un operador simple (en vez de compuesto, al constar de dos símbolos independientes: <<**-**>> y <<**>**>>). Se denomina operador de selección de miembro o también operador de selección de componente.

De modo visual el operador **P -> m** recuerda a una flecha que apunta del puntero p al objeto que contiene al miembro m.

P -> m significa lo mismo que (\*p).m

**Construcción de una lista**

Un algoritmo para la creación de una lista enlazada requiere de los siguientes pasos:

1. Declarar el tipo de dato y el puntero de cabeza o primero.
2. Asignar memoria para un elemento del tipo definido anteriormente utilizando el operador new.
3. Crear iterativamente el primer elemento (cabeza) y los elementos sucesivos de una lista enlazada simple.
4. Repetir hasta que no haya mas entrada para el elemento.

Para analizar las operaciones de una lista se deben crear tres elementos que permitan implementar la lista en enlazada simple:

**Clase nodo:** esta clase representa un nodo individual en la estructura de datos y contiene información sobre el valor almacenado en el nodo y punteros a los nodos vecinos.

**Clase lista:**  esta clase representará las operaciones que se pueden realizar con las listas.

**Método main:** método principal en donde se creará una instancia de la lista para acceder a sus operaciones.