Listas Enlazadas

Nivel lógico

Una lista es una colección de elementos homogéneos con una relación lineal entre ellos. El orden de los elementos en la lista afecta a su función de acceso.

Operaciones sobre listas ordenadas cuya función de acceso es que la lista está ordenada desde el elemento más pequeño al mayor:

- Crear_lista: inicializa lista a vacío.
- Insertar: añade un elemento a la lista.
- Quitar: suprime el elemento especificado de la lista.
- ImprimeLista: imprime todos los elementos de la lista.
- ListaVacía: operación booleana que indica si la lista está vacía.
- ListaLLena: operación booleana que indica si la lista está llena.

Representación de listas

Para representar una lista se puede usar una estructura secuencial o enlazada.

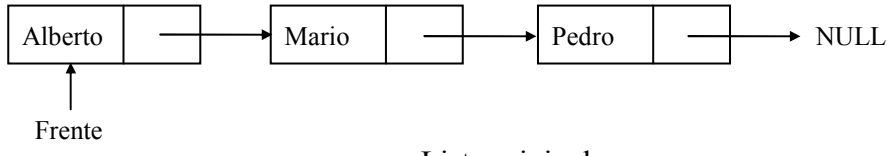
- Una lista secuencial es aquella en la que la colocación física de los elementos coincide con su orden lógico. Los elementos se colocan en posiciones secuenciales en la estructura. Ejm: Arrays.
- Una lista enlazada es aquella en la que el orden lógico de los elementos no es necesariamente equivalente a su colocación física. Para ello se usa un campo enlace explícito en cada elemento.

La ventaja de usar listas enlazadas es que permiten añadir o suprimir elementos manteniendo los elementos existentes en su lugar original, evitando movimiento de datos. Para ello, sólo se cambian los enlaces que establecen el orden de los elementos.

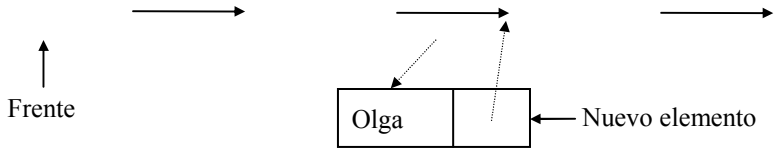
Nivel de implementación

En general hay dos formas:

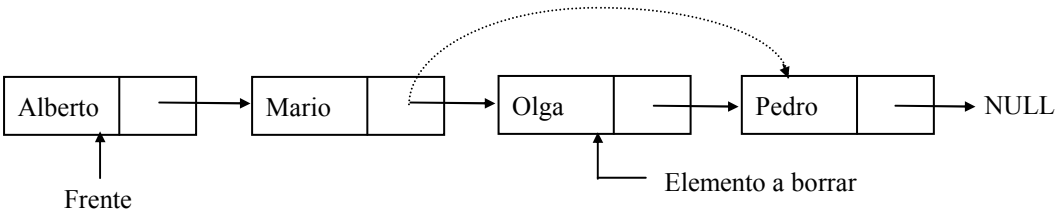
Usando variables dinámicas mediante punteros. En esta representación se puede considerar cada miembro de la lista como una unidad autocontenida (nodo) con un campo para la información y otro campo que almacena un puntero al siguiente elemento (sucesor). Como resultado un programa puede construir y procesar listas de tamaño arbitrario limitado unicamente por la memoria disponible.



Lista original



Inserción de un elemento



A continuación se muestra el código de implementación de una lista de enteros:

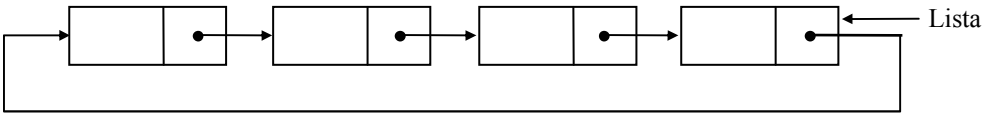
```
#include <stdio.h>
#include <alloc.h>
#include "lista.h"

struct lista {
int info;
struct lista *sig;
};

struct lista *frente=NULL;
int imprime_lista()
{
struct lista *temp;
int i=1;

temp=frente;
while (temp!=NULL)
 {
printf("Nodo %d: %d\n",i,temp->info);
i++;
temp=temp->sig;
 }
return(OK);
}

int crea_lista()
{
frente=NULL;
return(OK);
}

int lista_vacia()
{
if (frente==NULL)
return(LISTA_VACIA);
```

```
return(OK);
}

int inserta(int info)
{
struct lista *nuevo;
struct lista *ptr;

if((nuevo=(struct lista *)malloc(sizeof(struct lista)))==NULL)
return(SIN_MEMORIA);
nuevo->info=info;
nuevo->sig=NULL;
if (frente==NULL)
frente=nuevo;
else
if (info<(frente->info))
  {
nuevo->sig=frente;
frente=nuevo;
  }
else
  {
ptr=frente;
while(ptr->sig!=NULL)
if (info>=ptr->sig->info)
ptr=ptr->sig;
else
break;
nuevo->sig=ptr->sig;
ptr->sig=nuevo;
  }
return(OK);
}

int suprime(int info)
{
struct lista *actual, *anterior;
actual=frente;
anterior=NULL;

while(actual->info!=info)
 {
anterior=actual;
actual=actual->sig;
 }
if (anterior==NULL)
frente=frente->sig;
else
anterior->sig=actual->sig;
```

```
free(actual);
return(OK);
}
```

Listas circulares enlazadas

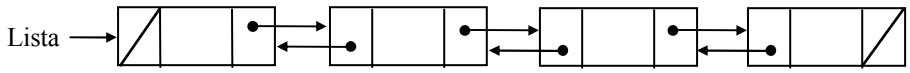
Son listas en la que cada nodo tiene un sucesor y el "último" elemento apunta al "primer elemento".



Una aplicación de la lista circular es usarla para representar una cola FIFO.

Listas doblemente enlazadas

Son listas en la que cada nodo va enlazando con su sucesor y su predecesor.



Con esta representación se facilitan enormemente las operaciones sobre listas.