**LinkList**

primero () {

si es inicio

entonces retorna los datos;

si no me retorna un 0;

}

moverse (solicito un entero posicion) {

Declare un Nuevo nodo temporal y le asigno el inicio;

for(recorre hasta la posicion comenzando en 1){

a temporal le asigno los datos

}

Retorno el temporal

}

bool LinkList::insertar(solicito un objeto y una posicion ) {

si la posición es menor a 1 o es mayor que el FIN(L) de elementos) {

entonces returna false;

} sino comienza lo bonito de agregar {

Si la posición ingresada es igual a 1 y los datos no están vacios {

Entonces creo un nuevo nodo temporal

Luego al cambio el dato por el nuevo dato

Y cambio el siguiente por el inicio

Verifico si incio es nulo

Entonces cambio el anterior por el incio por el temporal

}

Asigno el temporal al inicio

A numero de elementos le sumo uno

Y retorno un true

} sino {

Si la posición es igual a FIN(L) de elementos {

Creo dos nuevos nodos temporales; al primero le asigno moverse en la posición ingresada y el segundo lo creo vacio

Con el temporal2 cambio los datos existentes por el nuevo

El tempora1 lo uso para cambiar el siguiente por el temporal2

Luego cambio el anterior por el temporal1

Al numero de elementos le sumo 1

Y retorno true;

} sino {

Creo 3 nodos temporales nuevos al primero le asigno moverse en la posición ingresada el segundo lo creo vacio y el tercer temporal le asigno a moverse pero en la posición ingresada mas 1

Luego el temporal2 lo utilizo para cambiar el dato por el dato ingresado

Luego el temporal2 lo utilizo para cambiar el anterior por el temporal1

También cambio el siguiente por el temporal3

Y el anterior lo cambio por el temporal 2

Y al número de elementos le sumo 1

Y retorno true;

}

}

}

}

anular() {

eliminino inicio y le asigno un nulo

mientras inicio este vacio {

creo un nodo temporal y le asigno el inicio

A inicio le asigno el siguiente

si (datos esta vacio) {

elimino los datos del temporal

luego le asigno vacio a los datos

y cambio el siguiente por un vacio

}

Elimino el temporal para liberar memoria

}

Y al numero de elementos le asigno 0

}

Retorno un entero localizar (solicito un objeto) {

Creo un entero que me maneje la posición y la inicializo en 1

Creo un nuevo nodo temporal

mientras ((temporal no este vacío NULL) y (la posición sea menor o igual que el número de elementos)) {

entonces

si (temporal->getDatos()->igual(o)) {

retorno la posición

} sino {

Posición más 1;

}

A temporal le asigno el siguiente

si(inicio) {

elimino inicio;

luego para asegurar le asigno a inicio vacio o un nulo

y al numero de elementos le asigno 0

}

}

Y retorno -1;

}

Retorno un Objeto\* recuperar(int posicion) {

Si la posición es igual a 1{

Entonces returno primero();

} sino si (posicion es mayor o igual a 2 y la posicion es menor o igual al numero de elementos ) {

Entonces creo un nodo temporal y le asigno moverse en la posición ingresada y retorno el dato en esa posicion

} sino {

Entonces retorno 0;

}

}