	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	M.C. JOSÉ MAURICIO MATAMOROS DE MARIA Y CAMPOS
Asignatura:	ESTRUCTURA DE DATOS Y ALGORITMOS I
Grupo:	12
No de Práctica(s):	08
Integrante(s):	ARELLANES CONDE ESTEBAN
No. de Equipo de cómputo empleado:	16
No. de Lista o Brigada:	01
Semestre:	2022-2
Fecha de entrega:	25 abril, 23:59
Observaciones:	

CALIFICACIÓN: _____

Práctica 8. Listas doblemente ligadas

Estructuras de Datos y Algoritmos I

Autor: Arellanes Conde Esteban

1. Objetivo

El alumno revisará las definiciones, características, procedimientos y ejemplos de la estructura lineal lista doblemente ligada tanto en sus versiones de extremos abiertos como circulares para poder comprenderlas a fin de ser capaz de implementarlas.

2. Introducción

Una lista ligada es una estructura de datos que tiene un orden lineal de sus elementos, esta a diferencia de la formada por arreglos tiene apuntadores que determinan su orden apuntando al elemento siguiente. Si la lista es simplemente ligada omitimos el miembro de la estructura "prev" (recordando que una lista puede tener atributos de previo, siguiente y una "llave" que representa al valor contenido del elemento), si es circular el miembro último debe apuntar como siguiente al primero de la lista, si eademás es ordenada puede tener como máximo al último elemento y mínimo al primero.

3. Desarrollo de la práctica:

Modificando el código ‘‘`prog1.c`’’ para generar un programa que permita recorrer la lista generada en un bucle infinito en ambos sentidos.

¿Qué salida produjo el programa? [1 punto]: La salida que produjo el programa fue la siguiente:

```
List size: 84
List size: 81
List size: 83

L[0:5]: ... <->887 <->778 <->916 <->794 <->336 <->...
L[10:20]: ... <->28 <->691 <->60 <->764 <->927 <->541 <->427 <->173 <->737 <->212
<->369 <->568 <->430 <->783 <->531 <->863 <->124 <->68 <->55 <->136 <->...
L[-5:0]: ... <->...
L[-5:-5]: ... <->88 <->751 <->44 <->365 <->435 <->...
```

¿Cómo se implementó la función insert? Anote y explique su código a continuación [2 puntos]: Para implementar la función insert lo que hicimos fue primero crear un nodo, comprobar si el

elemento se va a insertar al principio o al final de la lista para enlazar el último elemento con el primero y mantener la lista circular previamente formada. En caso contrario, buscamos el lugar en el que se va a insertar el elemento y desconectamos el elemento siguiente para conectar al elemento que queremos insertar y aumentar la lista.

```
void insert(list* l, int ix, int x){
 node* n = (node*)calloc(1, sizeof (node));
 n->value = x;

 if(ix == 0){
 preppend(l, x);
 return;}

 node* m = l->first;
 for(;ix <0; ++ix) m = m->prev;
 for(;ix >0; --ix) m = m->next;
 connect(m->prev, n);
 connect(n, m);
 l->count++; }
```

¿Cómo se implementó la función append? Anote y explique su código a continuación [2 puntos]:
De manera similar a la función anterior sólo que ahora insertamos por delante del primer elemento.

```
void append(list* l, int x){
 preppend(l, x);
 l->first = l->first->next;
}
```

¿Cómo se implementó la función delete? Anote y explique su código a continuación [2 puntos]:
La función delete para que funcione tiene que tener elementos la lista y por ende tener valores positivos, pero al ser circular también debería aceptar valores negativos para recorrer y eliminar en orden inverso. Lo que hacemos es, básicamente, desconectar el elemento de la lista. Sin embargo, existe un caso particular para el que este no es aplicable ya que si elemento que queremos eliminar de la lista es el primero tenemos que recorrer este al siguiente para indicar el fin de la lista.

```
void delete(list* l, int ix){
 node* m = l->first;
 for(; ix <0; ++ix) m = m->prev;
 for(; ix >0; --ix) m = m->next;
 connect(m->prev, m->next);
 if(m == l->first)
 l->first = m->next;
 --l->count;
 free(m);
 }
```

4. Cuestionario:

1. ¿Qué estructura de datos requiere el programa? Explique [1 punto]:
   El programa requería de una lista doblemente enlazada o ligada.

2. ¿Cuál es la diferencia entre una lista ligada y una lista doblemente ligada? Explique [1 punto]:
   En que una es lineal y la otra se puede ver como un anillo por decirlo de alguna manera.

3. ¿Cuál es la diferencia entre una lista doblemente ligada y una lista doblemente ligada circular? Explique [1 punto]:
   En que la segunda (lista doblemente ligada circular) apunta su último elemento al head o primer elemento en seguida y no en reversa como la lista doblemente ligada.

5. Conclusión y Referencias

5.1. Conclusión:

Una lista circular es una variación de la estructura de datos simple de lista y esta al ser doblemente ligada cumple con todas las funciones de inserción al frente, por detrás y al medio, eliminación al frente y por detrás junto con la operación de búsqueda pero algunas son de O(n) que en otras palabras significa que son lentas ya que en el peor de los casos habría que recorrer todos los elementos de la lista hasta llegar al último elemento.

5.2. Referencias:

- Thomas H. Cormen. (2009). Introduction to Algorithms 3e. United States of America: Massachusets Institute of Technology.
- Stephen R. Davis. (2015). Beginning Programming with C++ For Dummies. New Jersey, United States of America: John Wiley & Sons.
- Código completo prog1.c: https://onlinegdb.com/YZt054rsy