FACULTAD DE CIENCIAS EXACTAS, INGENIERÍA Y AGRIMENSURA ESCUELA DE CIENCIAS EXACTAS Y NATURALES DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN ESTRUCTURAS DE DATOS Y ALGORITMOS I

Práctica adicional - Listas

- 1. Las listas enlazadas implementadas con punteros tienen la flexibilidad de poder insertar elementos en cualquier parte de ellas sin mover los elementos anteriores ni posteriores, mientras que los arreglos no cuentan con esta flexibilidad.
 - a) Proponga una implementación similar a listas enlazadas, pero de longitud fija, utilizando arreglos, y que provea esta flexibilidad.
 - b) Agregue que al intentar realizar una operación que supera la longitud disponible, realoque la información en un arreglo con el doble de tamaño (puede ser de utilidad la función realloc).

Nota: Ver sección "Implantación con arreglo de listas", página 195, "Estructuras de Datos con C" Tenenbaum.

- 2. Suponga que está implementando una lista doblemente enlazada. Si en lugar de guardar punteros a los nodos previo y siguiente, guarda un xor de ambos punteros: ¿puede recorrer la lista en ambas direcciones? ¿Cómo? Defina en C la estructura correspondiente, ¿qué problemas puede encontrar? Enuncie posibles ventajas y desventajas de esta implementación de listas.
- **3.** Cree un programa que manipule listas de enteros y que acepte comandos desde la entrada estándar de la siguiente forma:

Comando	Argumentos	Resultado	Ejemplo
crear	lista	Crea una lista	crear 1
destruir	lista	Destruye una lista	destruir 1
imprimir	lista	Imprime el contenido actual de la lista	imprimir 1
agregar_final	lista, elem	agrega elem al final de la lista	agregar_final 1 42
agregar_inicio	lista, elem,	agrega elem al principio de la lista	agregar_inicio 1 42
agregar_pos	lista, elem, pos	agrega elem a la lista en la posición pos	agregar_pos 1 42 3
longitud	lista	imprime la longitud de la lista	longitud 1
concatenar	11, 12, 13	concatena l1 y l2 y crea l3 con el resultado	concatenar 1 2 3
eliminar	lista, pos	elimina de lista el elemento en la posición pos	eliminar 1 5
contiene	lista, elem	imprime "SI" si la lista contiene a elem, "NO" sino	contiene 1 42
indice	lista, elem	imprime las posiciones en las que está elem	indice 1 42
intersecar	11, 12, 13	crea l3 con la intersección de l1 y l2	intersecar 1 2 3
ordenar	lista	ordena los elementos de la lista de menor a mayor	ordenar 1

Aclaración: Cada línea tendrá sólo un comando.

4. Dada una lista enlazada, retornar el nodo donde comienza el ciclo, o la dirección nula en caso que no haya ninguno. Por ejemplo, para la lista:

$$L: \quad \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$$

Retornar la dirección correspondiente al nodo 3.

5. Dada una lista enlazada de números enteros, eliminar todas las sub secuencias que sumen cero y retornar la lista resultante. En los casos que se puedan elegir una subsecuencia que esté contenida en otra, elegir la más grande. Por ejemplo:

Práctica adicional - Listas Página 1

Entrada: $\rightarrow 6 \rightarrow -6 \rightarrow 8 \rightarrow 4 \rightarrow -12 \rightarrow 9 \rightarrow 8 \rightarrow -8$

Salida: $\rightarrow 9$

Entrada: $\rightarrow 4 \rightarrow 6 \rightarrow -10 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow -19 \rightarrow 10 \rightarrow -18 \rightarrow 20 \rightarrow 25$

Salida: $\rightarrow 20 \rightarrow 25$

6. Encontrar el sufijo en común más largo entre dos listas enlazadas. Por ejemplo:

Entrada:

$$\rightarrow w \rightarrow a \rightarrow l \rightarrow k \rightarrow i \rightarrow n \rightarrow g$$

$$\rightarrow l \rightarrow i \rightarrow s \rightarrow t \rightarrow e \rightarrow n \rightarrow i \rightarrow n \rightarrow g$$

Salida: $\rightarrow i \rightarrow n \rightarrow g$

7. Si dos listas enlazadas comparten un nodo, también compartirán toda la sublista a partir de ese nodo. Por ejemplo:

$$L_1: \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 7 \rightarrow 8 \rightarrow 9$$

$$L_2: \rightarrow 5 \rightarrow 6$$

Se pide entonces, dadas dos listas enlazadas, encontrar la intersección de las mismas, es decir, el nodo a partir del cual las dos listas siguen igual (o la dirección nula en caso de no tener nodos en común). Para el ejemplo anterior, se debería retornar la dirección del nodo 7.

8. Dada la lista enlazada:

$$L_0 \to L_1 \to \ldots \to L_{n-1} \to L_n$$

Reordenarla a la forma:

$$L_0 \to L_n \to L_1 \to L_{n-1} \to L_2 \to L_{n-2} \to \dots$$

Se debe reutilizar los nodos de la lista dada.

9. Implementar una función que determine si una lista enlazada es palíndrome. Por ejemplo:

Entrada: $\rightarrow k \rightarrow a \rightarrow y \rightarrow a \rightarrow k$

Salida: True

2