

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.

- Nota:* Ver sección “*Implantación con arreglo de listas*”, página 195, “*Estructuras de Datos con C*” Tenenbaum.

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     | l1, l2, l3       | 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     | l1, l2, l3       | 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           |

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:

Retornar la dirección correspondiente al nodo 3.

- 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:

$L_0 \rightarrow L_1 \rightarrow \dots \rightarrow L_{n-1} \rightarrow L_n$

Reordenarla a la forma:

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

Se debe reutilizar los nodos de la lista dada.

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

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

Salida: *True*