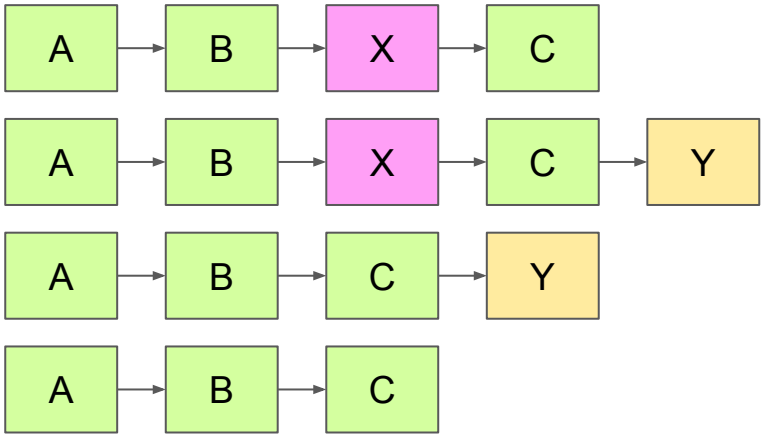
TDA LISTA ENLAZADA



TDA LISTA ENLAZADA

Una lista enlazada es una estructura de datos formada por nodos, donde cada nodo contiene un dato y un puntero al siguiente nodo. Permite insertar y eliminar elementos en cualquier posición sin mover el resto de los datos, ajustando sólo los punteros.







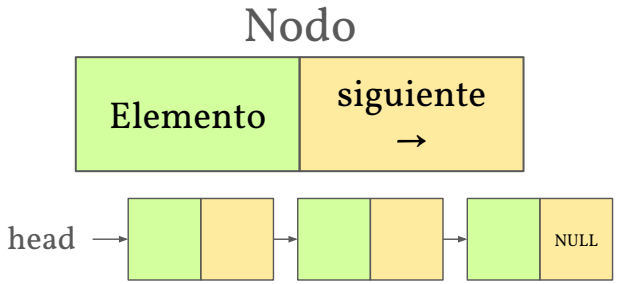
TDA LISTA: conjunto mínimo de operaciones

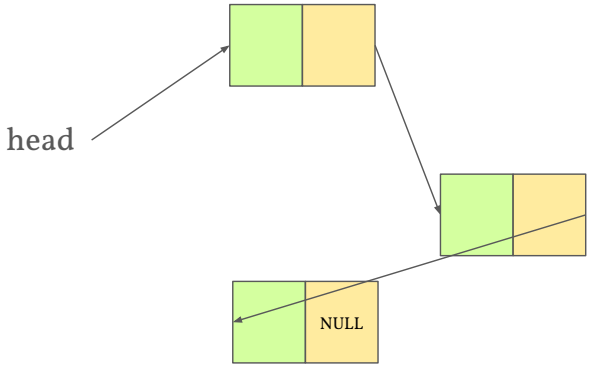
Para el caso de una lista el conjunto mínimo de operaciones es:

- `crear (create)`
- `insertar (insert)`
- `eliminar (delete)`
- `obtener (get)`
- `buscar (search)`
- `largo_lista (length)`
- `esta_vacia (is_empty)`
- `destruir (destroy)`

TDA LISTA ENLAZADA: ¿Cómo funciona internamente?

- La lista está compuesta por nodos.
  - Elemento
  - Puntero al nodo siguiente
- Para recorrer:
  - Empezas por el nodo apuntado por head y avanzas con el “siguiente” hasta encontrar un null.
- A diferencia de un arreglo, los nodos no están contiguos en memoria





TDA PILA: insertar()

Esta operación inserta un elemento en la lista al principio, al final o en cualquier posición intermedia de la misma.

Insertar un elemento en una lista enlazada consiste en crear un nuevo nodo y modificar los punteros para que quede conectado dentro de la secuencia.

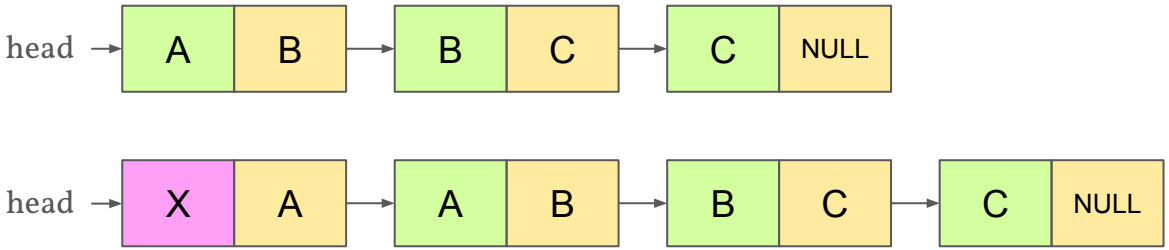
No hace falta mover otros elementos: solo se ajustan los enlaces.

Casos:

- Insertar al comienzo
- Insertar en el medio
- Insertar al final

TDA PILA: insertar()

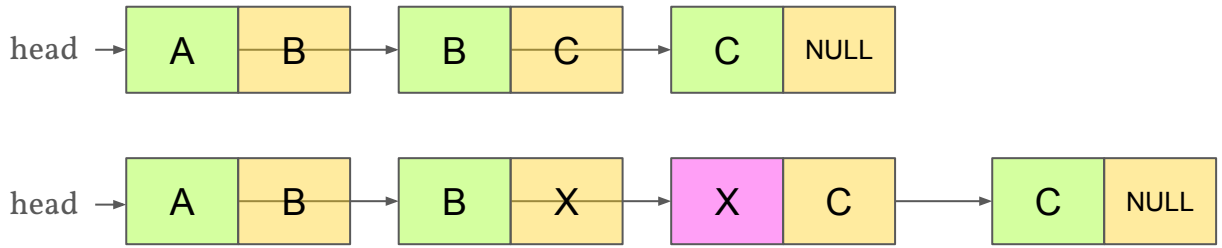
Al comienzo: insertar(X, o)



Ahora nuestro head va a apuntar al nodo X y este último va a apuntar a A.

TDA PILA: insertar()

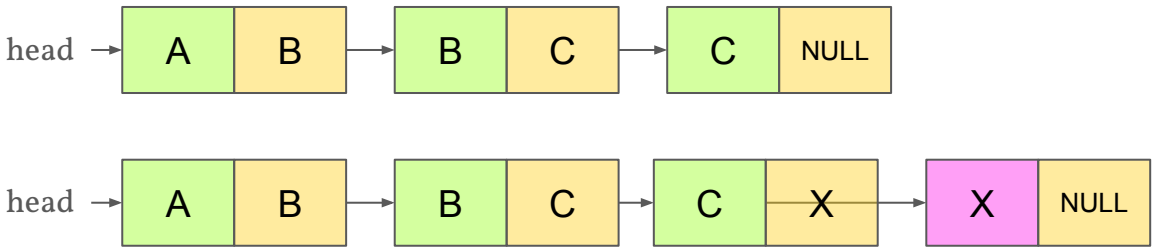
En el medio: insertar(X, 2)



Ahora nuestro nodo B va a apuntar al nuevo nodo X y este último es el que apunta a C

TDA PILA: insertar()

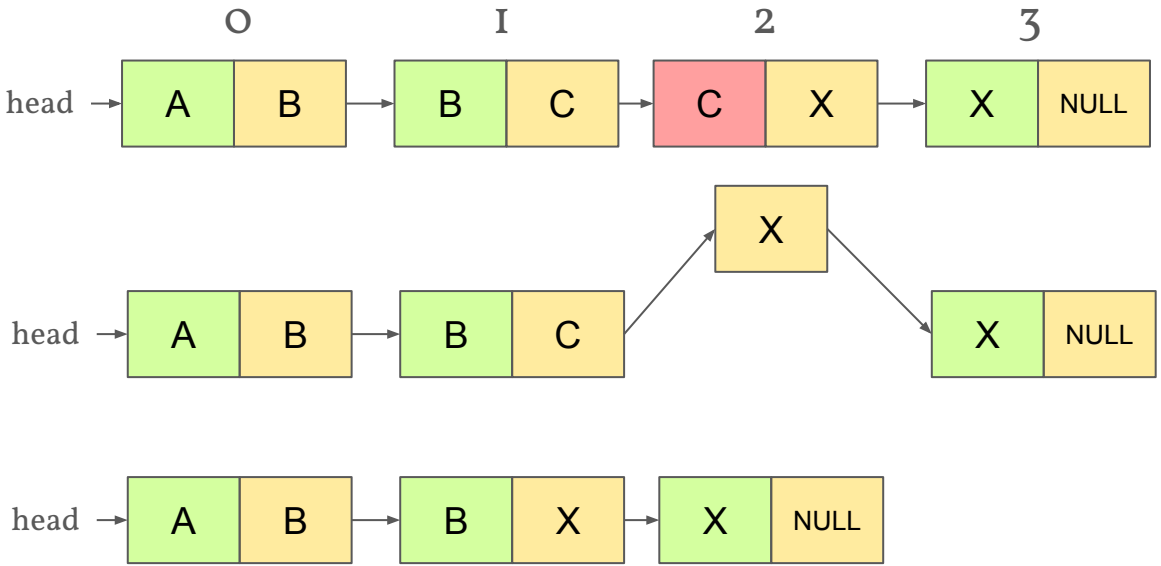
Al final: insertar(X, largo_lista())



Ahora nuestro nodo C va a apuntar al nuevo nodo X y este último es el que apunta a NULL

TDA LISTA ENLAZADA: eliminar()

Permite eliminar un elemento de la lista en cualquier posición

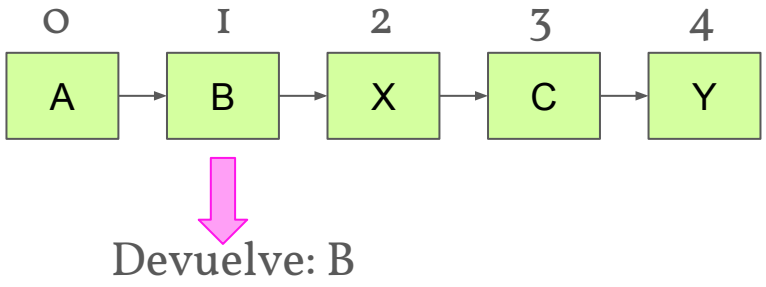


eliminar(2)

Antes de eliminar por completo el nodo C, tengo que guardarme el puntero a su siguiente porque sino pierdo la informacion y no puedo “reconectar” los nodos.

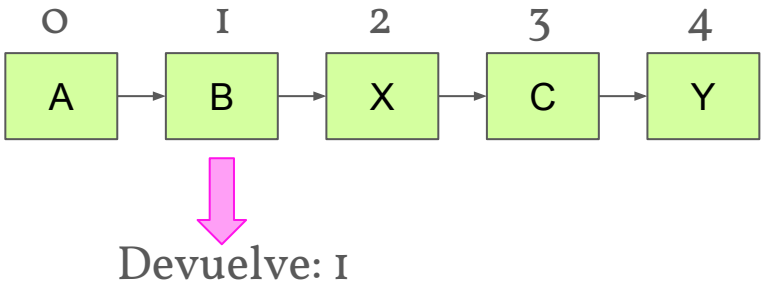
TDA LISTA ENLAZADA: obtener() vs buscar()

Obtener: Devuelve el elemento de la posición indicada.



obtener(1)

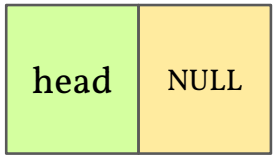
Buscar: Devuelve la posición en la que se encuentra el elemento buscado.



buscar(B)

TDA LISTA ENLAZADA: Otras operaciones

- **crear()**: head va a apuntar a NULL



- **largo_lista():** Devuelve cuántos elementos contiene la lista.
- **esta_vacia():** Valida si head apunta a NULL o no.
- **destruir():** LIBERA todo los recursos reservados al momento de la creación uno por uno. Siempre de atrás para adelante para no perder información en el camino.

Pros

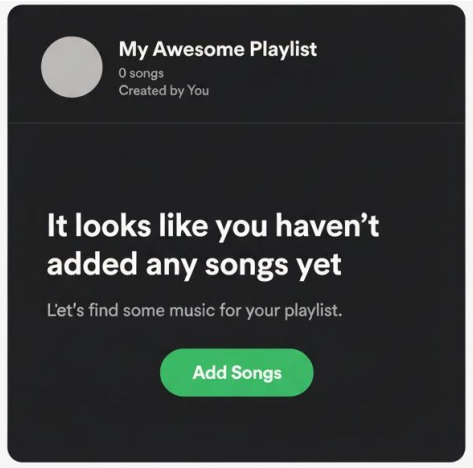
- Acceso a cualquier posición.
- Insertar/eliminar en el medio.
- No está limitada a un extremo.
- Estructuras dinámicas de tamaño flexible.

Contras

- Más compleja que una pila o cola.
- Acceso aleatorio es lento (hay que recorrer).
- Manejo de punteros más delicado (listas enlazadas).
- Precaución con memoria.

Ejemplo de uso: playlist

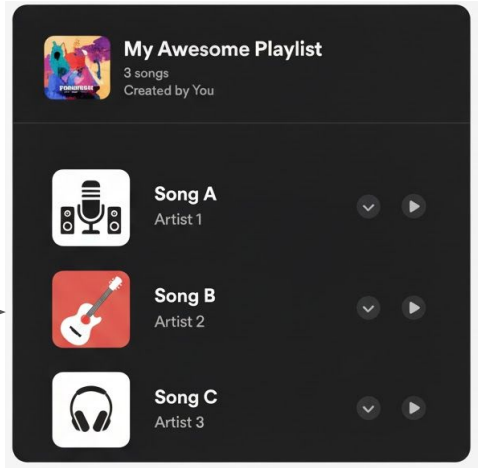
Crear()



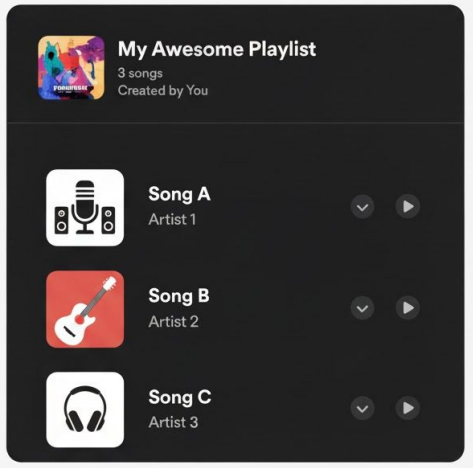
insertar(A, largo_lista)

insertar(B, largo_lista)

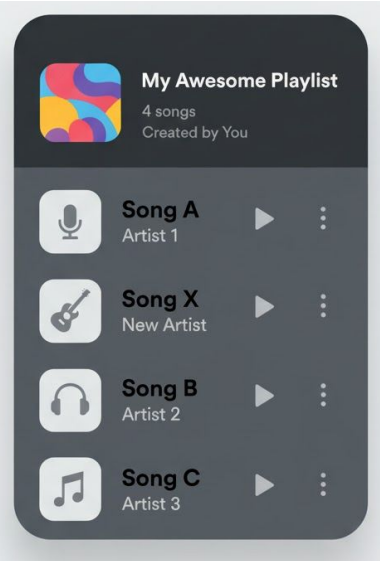
insertar(C, largo_lista)



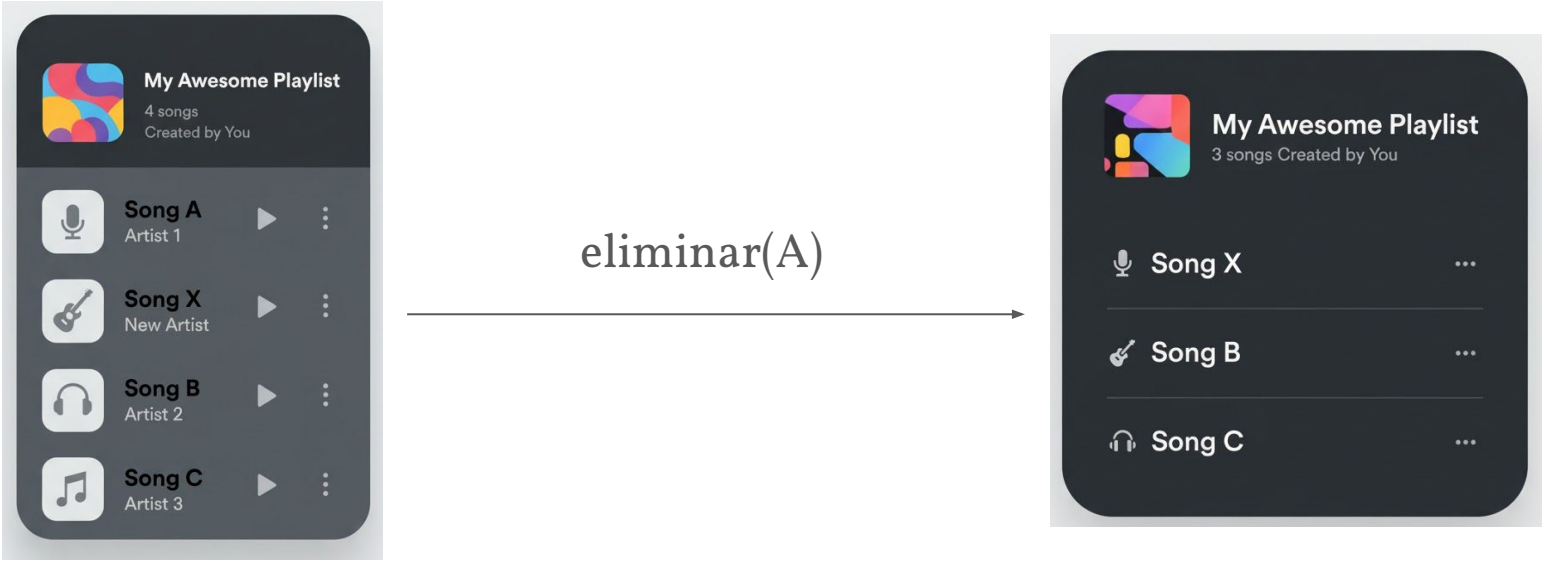
Ejemplo de uso: playlist



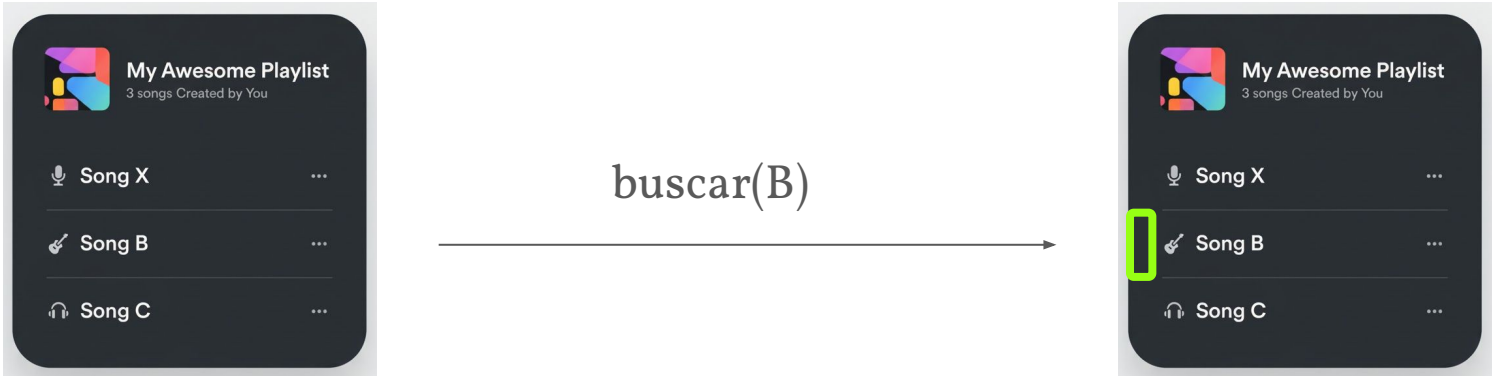
insertar(X, 1)

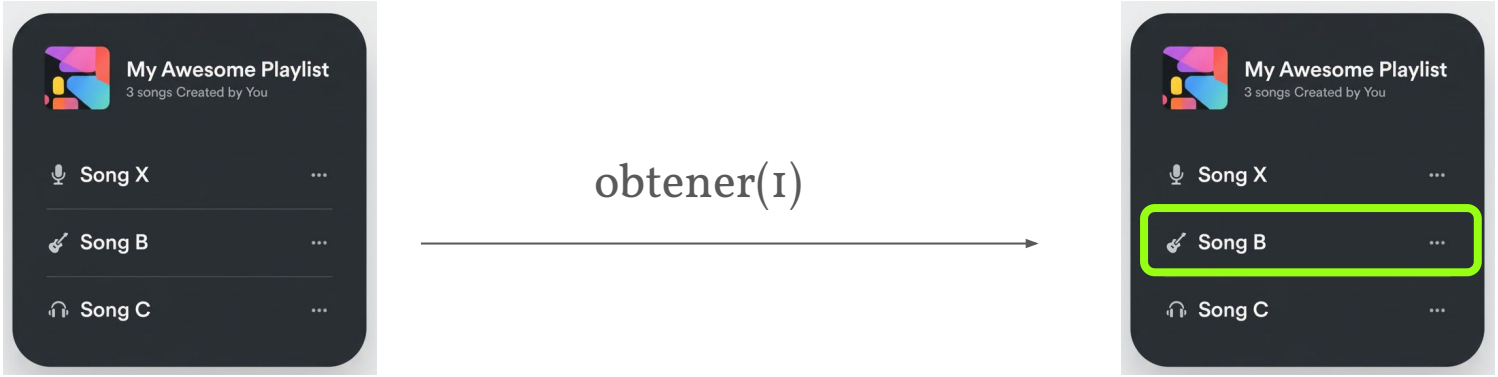


Ejemplo de uso: playlist

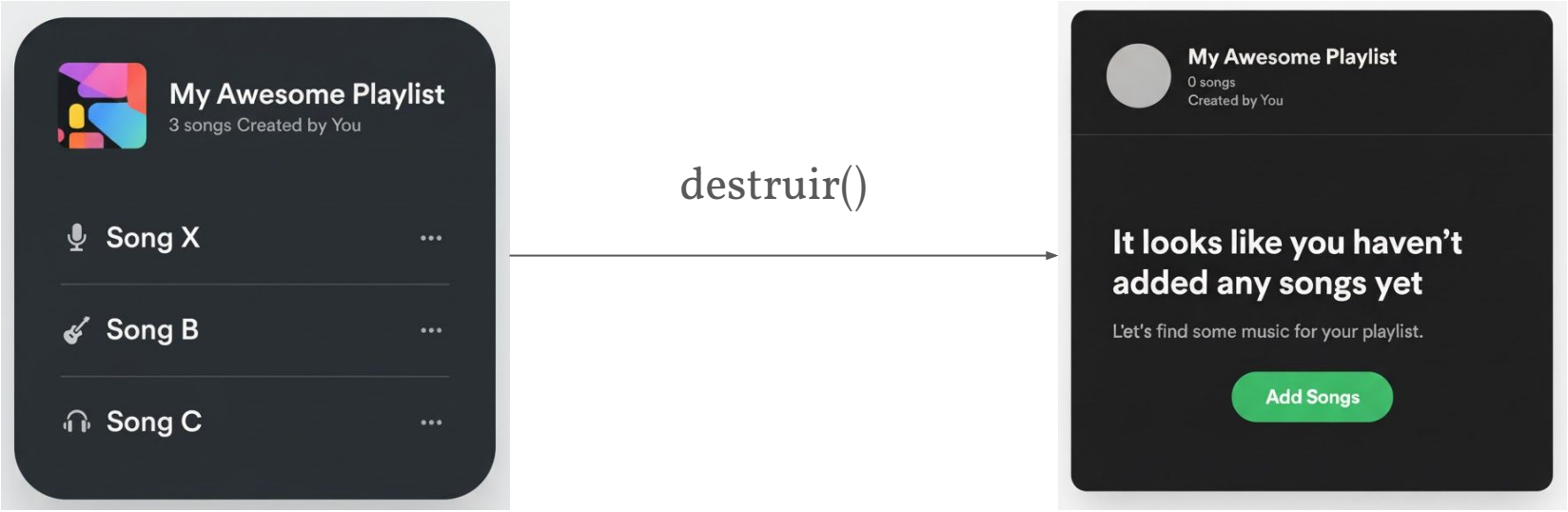


Ejemplo de uso: playlist





Ejemplo de uso: playlist



EJEMPLO