**Listas Doblementes Enlazadas**

**¿Que son?**

Según el foro GeeksforGeeks (2025): Una lista doblemente enlazada es una estructura de datos más compleja que una lista simple, pero ofrece varias ventajas. La principal es que permite un recorrido eficiente de la lista en ambas direcciones. Esto se debe a que cada nodo de la lista contiene un puntero al nodo anterior y otro al siguiente. Esto permite insertar y eliminar nodos de la lista de forma rápida y sencilla, así como un recorrido eficiente de la lista en ambas direcciones.

**Ventajas y Desventajas**

Según GeeksforGeeks (2025) las listas doblemente enlazadas presentan distintas ventajas y desventajas algunas ventajas son:

* Recorrido eficiente en ambas direcciones: Las listas doblemente enlazadas permiten un recorrido eficiente de la lista en ambas direcciones, lo que las hace ideales para aplicaciones que requieren inserciones y eliminaciones frecuentes.
* Fácil inserción y eliminación de nodos: La presencia de punteros tanto al nodo anterior como al siguiente facilita la inserción o eliminación de nodos de la lista, sin tener que recorrerla por completo.
* Se pueden usar para implementar una pila o una cola: Las listas doblemente enlazadas permiten implementar tanto pilas como colas, estructuras de datos comunes en programación.

También tienen algunas desventajas ya que al ser doblemente enlazadas se vuelven más complejas

* Más complejas que las listas enlazadas simples: Las listas doblemente enlazadas son más complejas que las listas enlazadas simples, ya que requieren punteros adicionales para cada nodo.
* Mayor consumo de memoria: Las listas doblemente enlazadas requieren más consumo de memoria que las listas enlazadas simples, ya que cada nodo almacena dos punteros en lugar de uno.

**Aplicaciones**

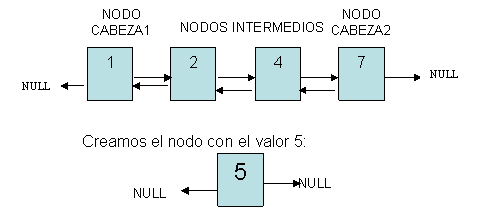
Existen diversas aplicaciones a las listas según GeekforGeeks algunas de las aplicaciones mas usuales son

* Implementación de la función de deshacer y rehacer en editores de texto.
* Implementación de caché donde se requiere la inserción y eliminación rápida de elementos.
* Gestión del historial del navegador para navegar entre las páginas visitadas.
* Aplicaciones de reproducción de música para gestionar listas de reproducción y navegar por las canciones de forma eficiente.
* Implementación de estructuras de datos como Deque (cola de doble extremo) para una inserción y eliminación eficiente en ambos extremos.

**Funcionamiento Interno**

Según GeeksforGeeks cada nodo de una lista doblemente enlazada contiene los datos que contiene, un puntero al siguiente nodo de la lista y un puntero al nodo anterior. Al enlazar estos nodos mediante los punteros "next" y "prev", podemos recorrer la lista en ambas direcciones (hacia adelante y hacia atrás), lo cual es una característica clave de una lista doblemente enlazada.

**Ejemplo**



class Nodo:

def \_\_init\_\_(self, dato):

self.dato = dato

self.anterior = None

self.siguiente = None

class ListaDoble:

def \_\_init\_\_(self):

self.cabeza = None

def agregar\_al\_final(self, dato):

nuevo = Nodo(dato)

if self.cabeza is None:

self.cabeza = nuevo

else:

actual = self.cabeza

while actual.siguiente:

actual = actual.siguiente

actual.siguiente = nuevo

nuevo.anterior = actual

def imprimir\_desde\_inicio(self):

actual = self.cabeza

while actual:

print(actual.dato, end=" ")

actual = actual.siguiente

print()

def imprimir\_desde\_final(self):

actual = self.cabeza

if not actual:

return

# Vamos al último nodo

while actual.siguiente:

actual = actual.siguiente

# Imprimir hacia atrás

while actual:

print(actual.dato, end=" ")

actual = actual.anterior

print()

GeeksforGeeks. (2025, February 19). *Doubly linked list tutorial*. GeeksforGeeks. https://www.geeksforgeeks.org/doubly-linked-list/