



LISTAS

Tipos, operaciones básicas y aplicaciones.

INTRODUCCIÓN

En C++, una lista es una estructura de datos lineal que permite almacenar elementos de forma ordenada, donde cada elemento (o nodo) contiene un valor y un puntero al siguiente (y, en listas doblemente enlazadas, también al anterior). Las listas facilitan operaciones dinámicas como inserciones y eliminaciones de elementos en cualquier posición, sin necesidad de reorganizar todos los elementos, siendo ideales para datos que cambian frecuentemente en tamaño o posición.

TIPOS DE LISTAS

Las listas son estructuras de datos lineales que permiten almacenar elementos de manera ordenada y acceder a ellos de forma secuencial.

- Cada nodo tiene un puntero que apunta al siguiente nodo.
- No permite recorrer la lista en ambos sentidos.
- Útil cuando no se conoce de antemano el número de elementos.

Enlazada Simple

- Cada nodo tiene dos punteros: uno al nodo siguiente y otro al nodo anterior.
- Permite recorrer la lista en ambas direcciones.
- Es útil cuando se necesita realizar inserciones o eliminaciones frecuentes en ambas direcciones de la lista.

Dblemente Enlazada

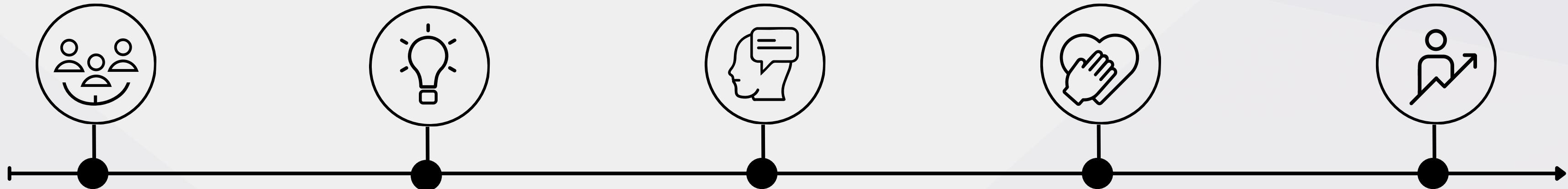
- En una lista enlazada circular, el último nodo apunta de vuelta al primer nodo.
- Puede ser tanto simple como doblemente enlazada.
- Útil para aplicaciones que requieren un acceso cíclico a los datos (por ejemplo, una lista de reproducción)

Lista Circulares

- Una lista donde cada nodo contiene una lista adicional.
- Útil en estructuras jerárquicas o aplicaciones que requieren listas anidadas

Lista Multinivel

Operaciones Básicas



INserción

- Inicio: Agregar un nuevo nodo al comienzo de la lista.
- Final: Agregar un nuevo nodo al final de la lista.
- Posición específica: Insertar un nodo en una posición dada dentro de la lista.

ELIMINACIÓN

- Inicio: Quitar el primer nodo de la lista.
- Final: Quitar el último nodo de la lista.
- Posición específica: Eliminar un nodo en una posición específica de la lista.

BÚSQUEDA

Localizar un nodo dentro de la lista basado en un criterio, como el valor almacenado en el nodo

RECORRIDO

Recorrer todos los elementos de la lista y realizar una operación en cada nodo, como imprimir los valores o aplicar un cálculo.

ACTUALIZACIÓN

Modificar el valor de un nodo en una posición específica.

APLICACIONES

IMPLEMENTACIÓN DE PILA Y COLA:

Una lista enlazada se puede usar para implementar estructuras de pila (LIFO) y cola (FIFO).

GESTIÓN DE PROCESOS EN SISTEMAS OPERATIVOS:

Los sistemas operativos utilizan listas circulares o doblemente enlazadas para gestionar procesos o tareas en ejecución.

EDITOR DE TEXTO

Las listas se usan para gestionar líneas de texto de forma eficiente, permitiendo inserciones y eliminaciones rápidas en cualquier parte del texto.



APLICACIONES

REPRESENTACIÓN DE GRAFOS

Las listas de adyacencia son una implementación común para representar grafos en algoritmos de grafos.

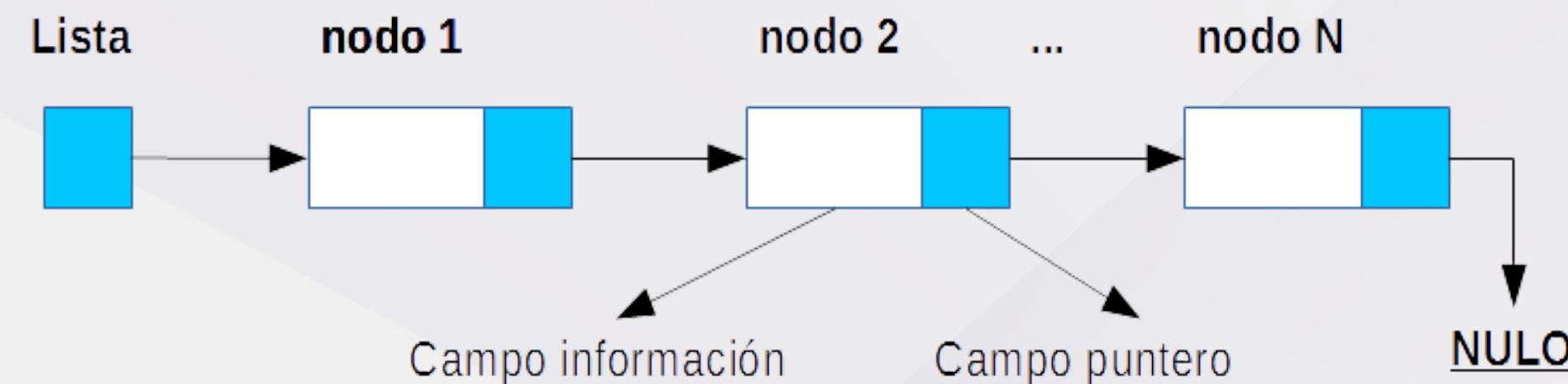
SIMULACIONES Y JUEGOS

Las listas son útiles para simular entidades en movimiento, como vehículos o personajes en videojuegos, permitiendo agregar o quitar elementos de forma dinámica.

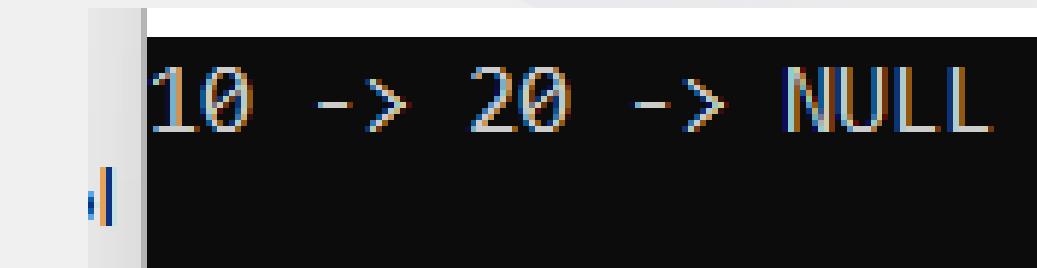
EJEMPLO

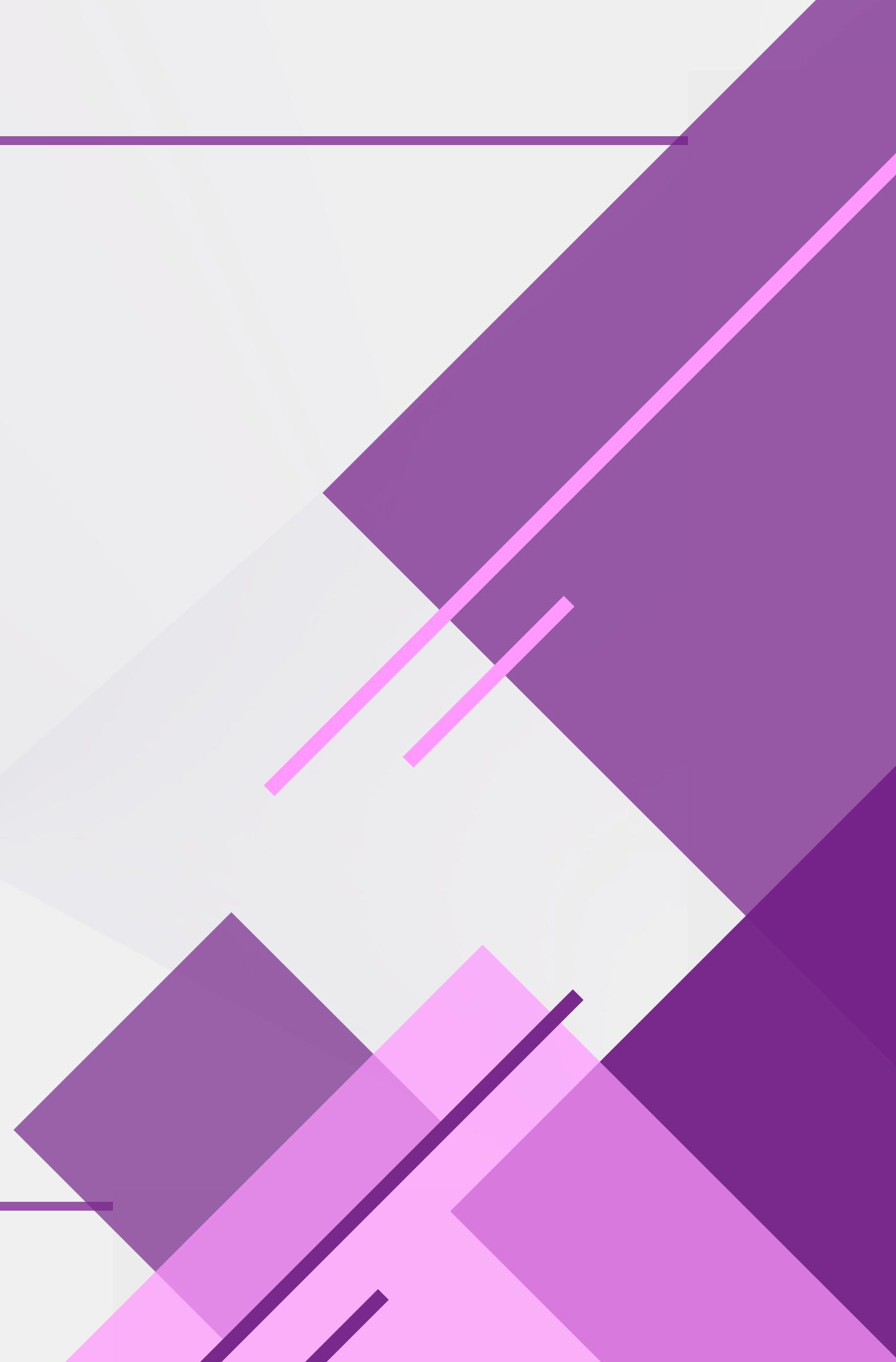
```
#include <iostream>
using namespace std;
// Definición de un nodo
struct Nodo {
    int valor;
    Nodo* siguiente;
};
int main() {
    // Crear un nodo y asignar valor
    Nodo* cabeza = new Nodo;
    cabeza->valor = 10;
    cabeza->siguiente = nullptr; // El siguiente
    nodo es nulo
    // Crear otro nodo y asignar valor
    Nodo* segundo = new Nodo;
    segundo->valor = 20;
    segundo->siguiente = nullptr;
    // Conectar el primer nodo con el segundo
    cabeza->siguiente = segundo;
    // Mostrar los valores de los nodos
    cout << cabeza->valor << " -> " <<
    cabeza->siguiente->valor << " -> NULL" <<
    endl;
    return 0;
}
```

Lista simplemente enlazada



(Imagen de creación propia)





**MUCHAS
GRACIAS**

