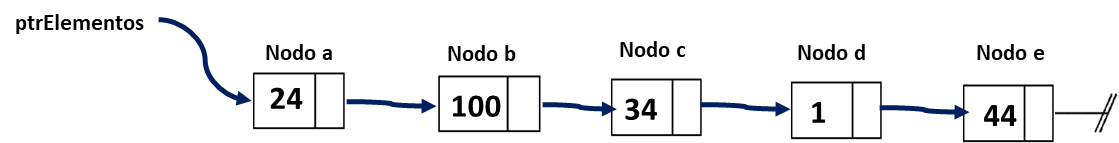
**Ejercicios TDA Lista**

1. Implemente una función que verifique si una lista es palíndroma o no, **LISTA IMPLEMENTADA CON PUNTEROS**.
2. Implemente un método que inserte un dato en una lista en la posición k de la lista, donde k en un numero entero, **LISTA IMPLEMENTADA CON PUNTEROS**.
3. Implemente un método que elimina el nodo que tengo el mayor número entero de la lista, **LISTA IMPLEMENTADA CON PUNTEROS**.
4. Implemente una función que devuelve una nueva lista sin los elementos repetidos de la lista original**, EJM:**

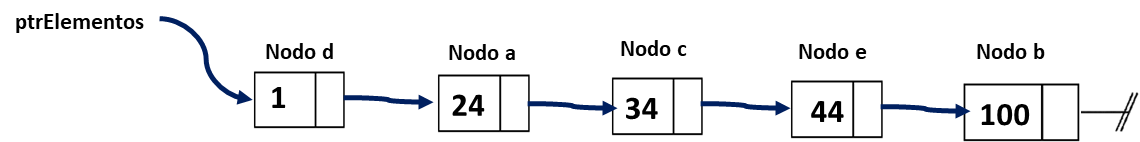
**Lista: 5,3,2,4,3,10,5,6 ----> Lista Nueva: 2,4,3,10,5,6**

**LISTA IMPLEMENTADA CON PUNTEROS**.

1. Implemente un método que ordene todos los elementos de la lista, de manera que no solo se intercambien los datos, sino también los enlaces de los nodos, **EJM:**



**LISTA ORDENADA**

****

**METODO QUE LES AYUDARA PARA MOSTRAR LAS DIRECCIONES DE LOS NODOS Y SUS DATOS:**

**void** ListaPtr::muestra\_nodos() {

cout << "ptrElementos --> ";

if (vacia()) {

cout << "NULL";

}

else {

NodoL\* aux = ptrElementos;

while (aux != NULL) {

cout << "Nodo direccion: " << &(\*aux);

cout << " | Nodo dato: " << aux->elemento;

if (aux->sig == NULL)

cout << " | Nodo sig: NULL";

else

cout << " | Nodo sig: " << &(\*aux->sig);

aux = aux->sig;

cout << endl << " ";

}

}

cout << endl;

}