



ESCUELA DE  
INGENIERÍA EN CIENCIAS Y SISTEMAS  
FACULTAD DE INGENIERÍA  
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



**Día, Fecha:**

Sábado, 15/04/2022

**Hora de inicio:**

19:00

# Introducción a la programación y computación 1 [D]

José Eduardo Morales García

# Introducción a las listas ligadas simples

- ▶ Son una estructura de datos utilizada para almacenar y acceder a una colección de elementos en orden secuencial. Cada elemento en la lista se representa mediante un nodo, el cual contiene un valor y un puntero al siguiente nodo en la lista.

# Explicación de su estructura

- ▶ A diferencia de los arreglos, en las listas ligadas simples, no se requiere que los elementos se almacenen en posiciones contiguas de memoria. Cada elemento se enlaza a su sucesor en la lista mediante un puntero. El último elemento de la lista tiene un puntero que apunta a **null**.

# Ventajas de su uso

- ▶ Las listas ligadas simples tienen varias ventajas. Una de las más importantes es que permite la inserción o eliminación de elementos de manera eficiente. Además, las listas ligadas son dinámicas, lo que significa que su tamaño puede cambiar durante la ejecución del programa. Otra ventaja es que, a diferencia de los arreglos, las listas ligadas no tienen un límite máximo de elementos que pueden contener.

# Implementación de una lista ligada simple en Java


- ▶ Para implementar una lista ligada simple en Java, necesitamos dos clases: **Nodo** y **ListaLigadaSimple**.
- ▶ La clase **Nodo** se utiliza para representar cada elemento de la lista. Cada nodo contiene un valor y un puntero al siguiente nodo en la lista.
- ▶ La clase **ListaLigadaSimple** contiene un puntero a la cabeza de la lista, es decir, el primer nodo de la lista.

# Agregar elementos a la lista y mostrarlos en consola

- ▶ Para agregar elementos a la lista, creamos un nuevo nodo y lo agregamos al final de la lista. Para hacer esto, primero verificamos si la lista está vacía. Si la lista está vacía, entonces establecemos el nuevo nodo como la cabeza de la lista. Si la lista no está vacía, recorremos la lista hasta encontrar el último nodo y lo enlazamos al nuevo nodo.
- ▶ Para mostrar los elementos de la lista en consola, recorremos la lista desde la cabeza hasta el último nodo y mostramos el valor de cada nodo.

# Ejemplo de uso de una lista ligada simple

- ▶ Primero, creamos una instancia de la clase `ListaLigadaSimple` para representar nuestra lista. Luego, agregamos algunos números a la lista utilizando el método `agregar`. Finalmente, mostramos los elementos de la lista en consola utilizando el método `mostrar`.
- ▶ En la consola, podemos ver que los números se muestran en el orden en que se agregaron a la lista, lo que demuestra que la lista mantiene el orden secuencial de los elementos.



```
public class Nodo {  
    int valor;  
    Nodo siguiente;  
  
    public Nodo(int valor) {  
        this.valor = valor;  
        this.siguiente = null;  
    }  
}
```



```
public class ListaLigadaSimple {
    Nodo cabeza;

    public ListaLigadaSimple() {
        this.cabeza = null;
    }

    public void agregar(int valor) {
        Nodo nuevoNodo = new Nodo(valor);
        if (cabeza == null) {
            cabeza = nuevoNodo;
        } else {
            Nodo ultimo = cabeza;
            while (ultimo.siguiete != null) {
                ultimo = ultimo.siguiete;
            }
            ultimo.siguiete = nuevoNodo;
        }
    }
}
```

```
public void imprimir() {
    Nodo actual = cabeza;
    while (actual != null) {
        System.out.println(actual.valor);
        actual = actual.siguiete;
    }
}
```

# Ligadas doblemente

- ▶ son una estructura de datos similar a las listas ligadas simples, pero con la capacidad de acceder tanto al elemento anterior como al siguiente en la lista. Cada elemento en la lista se representa mediante un nodo, el cual contiene un valor y punteros al nodo anterior y siguiente en la lista.

# Explicación de su estructura

- ▶ Cada nodo en una lista ligada doblemente tiene dos punteros, uno que apunta al nodo anterior y otro que apunta al nodo siguiente. Al igual que las listas ligadas simples, los nodos no tienen que estar almacenados en posiciones contiguas de memoria y se enlazan mediante punteros. El primer nodo en la lista tiene un puntero al nodo anterior que apunta a **null**, mientras que el último nodo en la lista tiene un puntero al nodo siguiente que apunta a **null**.

# Ventajas de su uso

- ▶ Las listas ligadas doblemente tienen algunas ventajas sobre las listas ligadas simples, como la capacidad de acceder tanto al elemento anterior como al siguiente en la lista, lo que facilita la inserción y eliminación de elementos. Además, la capacidad de acceder tanto al elemento anterior como al siguiente también permite la implementación de operaciones de búsqueda más eficientes.

# Implementación de una lista ligada doblemente en Java

Para implementar una lista ligada doblemente en Java, necesitamos dos clases: `Nodo` y `ListaLigadaDoble`.

La clase `Nodo` se utiliza para representar cada elemento de la lista. Cada nodo contiene un valor y punteros al nodo anterior y siguiente en la lista.

La clase `ListaLigadaDoble` contiene un puntero a la cabeza de la lista, es decir, el primer nodo de la lista.

# Agregar elementos a la lista y mostrarlos en consola

- ▶ Para agregar elementos a la lista, creamos un nuevo nodo y lo agregamos al final de la lista. Para hacer esto, primero verificamos si la lista está vacía. Si la lista está vacía, entonces establecemos el nuevo nodo como la cabeza de la lista. Si la lista no está vacía, recorremos la lista hasta encontrar el último nodo y lo enlazamos al nuevo nodo.
- ▶ Para mostrar los elementos de la lista en consola, recorremos la lista desde la cabeza hasta el último nodo y mostramos el valor de cada nodo.

# Agregar elementos a la lista y mostrarlos en consola

- En conclusión, las listas ligadas doblemente son una estructura de datos útil cuando necesitamos acceder a elementos anteriores y posteriores en una lista. Su implementación en Java es sencilla y se compone de una clase **Nodo** y una clase **ListaLigadaDoble**. Podemos agregar elementos a la lista y mostrarlos en consola mediante la creación de nuevos nodos y estableciendo los punteros apropiados.

```
public class Nodo {  
    public int valor;  
    public Nodo anterior;  
    public Nodo siguiente;  
  
    public Nodo(int valor) {  
        this.valor = valor;  
        this.anterior = null;  
        this.siguiente = null;  
    }  
}
```



```
public class ListaLigadaDoble {
    private Nodo cabeza;

    public ListaLigadaDoble() {
        this.cabeza = null;
    }

    public void agregar(int valor) {
        Nodo nuevoNodo = new Nodo(valor);
        if (this.cabeza == null) {
            this.cabeza = nuevoNodo;
        } else {
            Nodo nodoActual = this.cabeza;
            while (nodoActual.siguiente != null) {
                nodoActual = nodoActual.siguiente;
            }
            nodoActual.siguiente = nuevoNodo;
            nuevoNodo.anterior = nodoActual;
        }
    }
}
```

```
public void mostrar() {
    Nodo nodoActual = this.cabeza;
    while (nodoActual != null) {
        System.out.println(nodoActual.valor);
        nodoActual = nodoActual.siguiente;
    }
}
```

```
public class Main {  
    public static void main(String[] args) {  
        ListaLigadaDoble lista = new ListaLigadaDoble();  
        lista.agregar(5);  
        lista.agregar(10);  
        lista.agregar(15);  
        lista.mostrar();  
    }  
}
```



Parte practica