Universidad Nacional Mayor de San Marcos
Universidad del Perú. Decana de América

Estructura de Datos

Semana 3

Universidad Nacional Mayor de San Marcos
Universidad del Perú. Decana de América

Logro de la sesión

Al finalizar la sesión, el estudiante:

- **Utiliza en forma crítica las listas enlazadas para la solución de problemas e implementa programas utilizando listas enlazadas.**

Estructuras de datos lineales

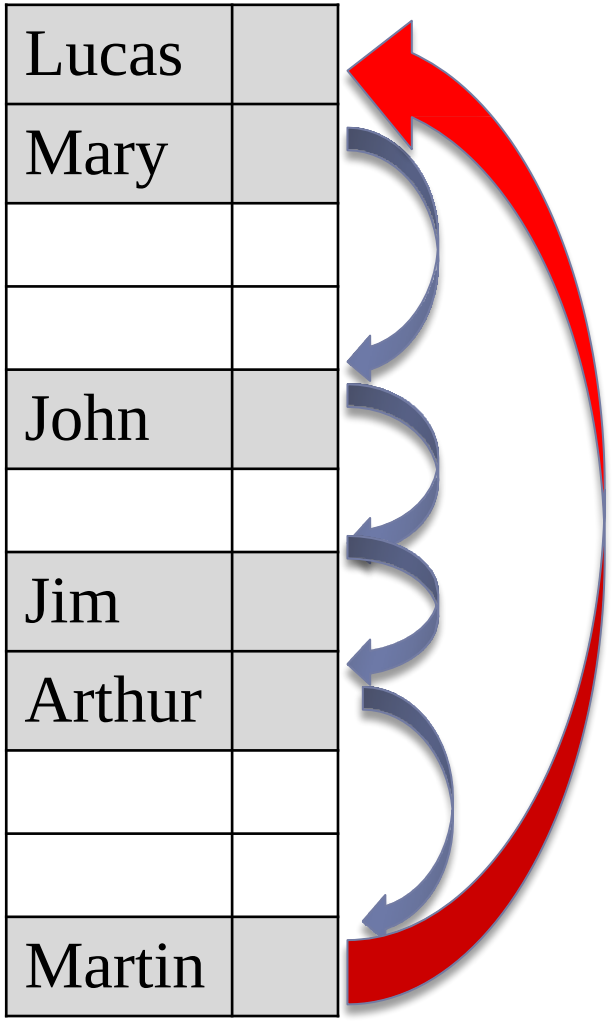
01

Implementación de un TAD lineal utilizando una estructura dinámica

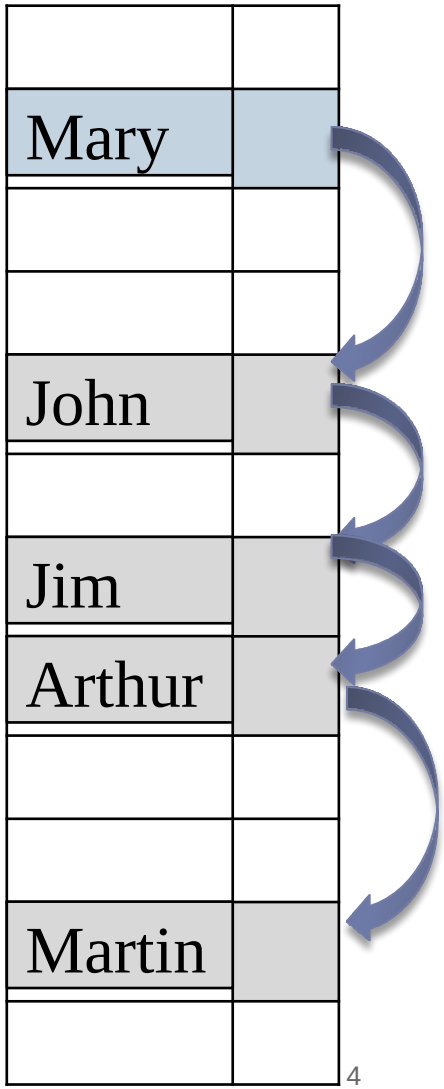
Estructuras de datos lineales

- **Estructura dinámica**

Los espacios en la memoria permiten que las órdenes físicas y lógicas puedan ser diferentes



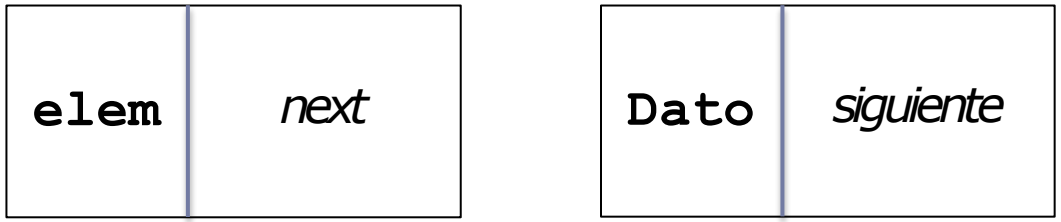
Cada ubicación no solo almacena (una referencia a) un objeto, sino también la referencia (dirección) a su sucesor en la Lista



Estructuras de datos lineales

- Necesitamos definir una clase, **Nodo**, para representar cada ubicación. La clase Nodo tiene dos atributos:
  - **Dato o elem:** es la referencia a un elemento almacenado en la Lista. Es decir, almacena la dirección de memoria donde se almacena el elemento. El tipo de datos de elem debe ser del mismo tipo que los elementos de la Lista
  - **Next o siguiente:** es la referencia al nodo que almacena el siguiente elemento en la Lista. Su tipo de datos debe ser **Nodo**

Nodo



Estructuras de datos lineales

- **Especificación formal:**
  - Secuencia de elementos **{a1,a2,…,an}** donde cada elemento tiene un único predecesor (excepto el primero que no tiene predecesor) y un único sucesor (excepto el último que no tiene sucesor).

  - Las operaciones básicas de un TAD Lista son:
    - **Agregar** un elemento a la Lista
    - **Eliminar** un elemento de la Lista
    - **Consultar** un elemento de la Lista

Estructuras de datos lineales

- **Especificación formal (operaciones de agregar):**

```
public interface IList {

    public void addFirst(String newElem);

    public void addLast(String newElem);

    public void insertAt(int index, String newElem);
```

Estructuras de datos lineales

- **Especificación formal (operaciones de consulta):**

```
public boolean isEmpty();

public boolean contains(String elem);

public int getSize();

public int getIndexOf(String elem);

public String getFirst();

public String getLast();

public String getAt(int index);

public String toString();
```

Estructuras de datos lineales

- **Especificación formal (operaciones de borrado):**

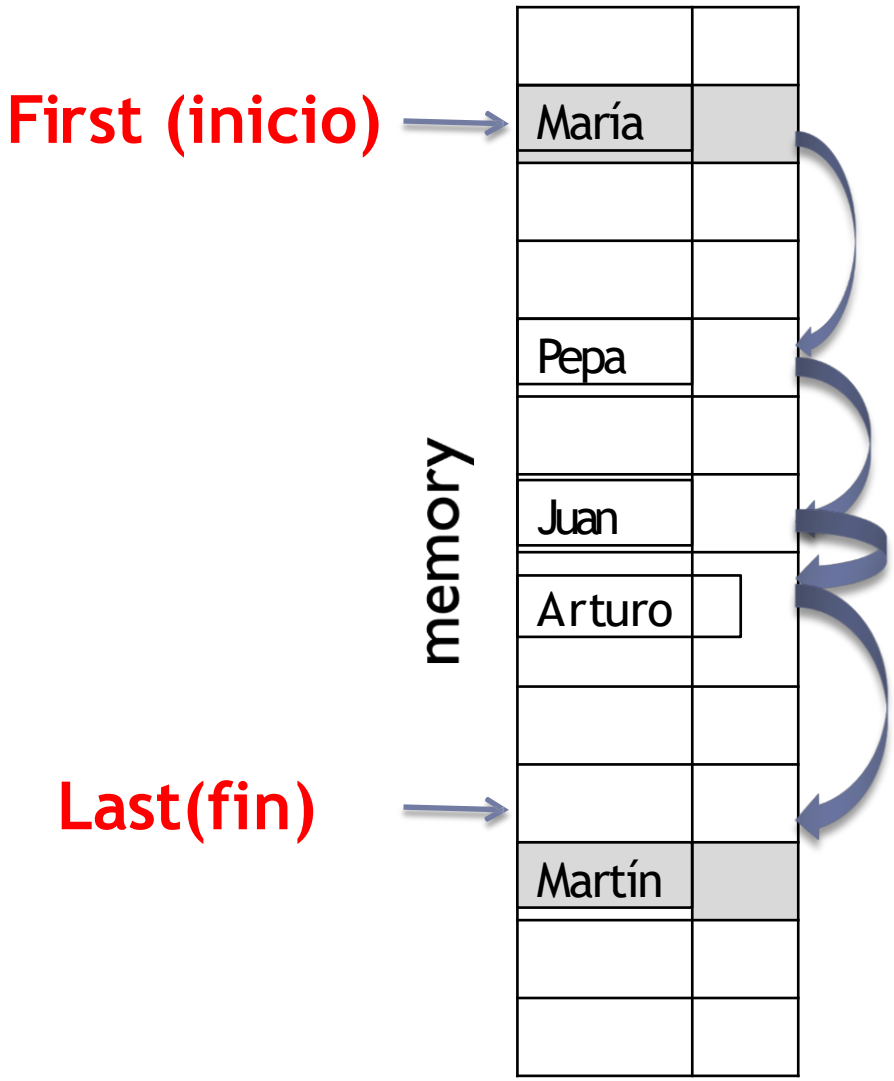
```
public void removeFirst();

public void removeLast();

public void removeAll(String elem);

public void removeAt(int index);
```

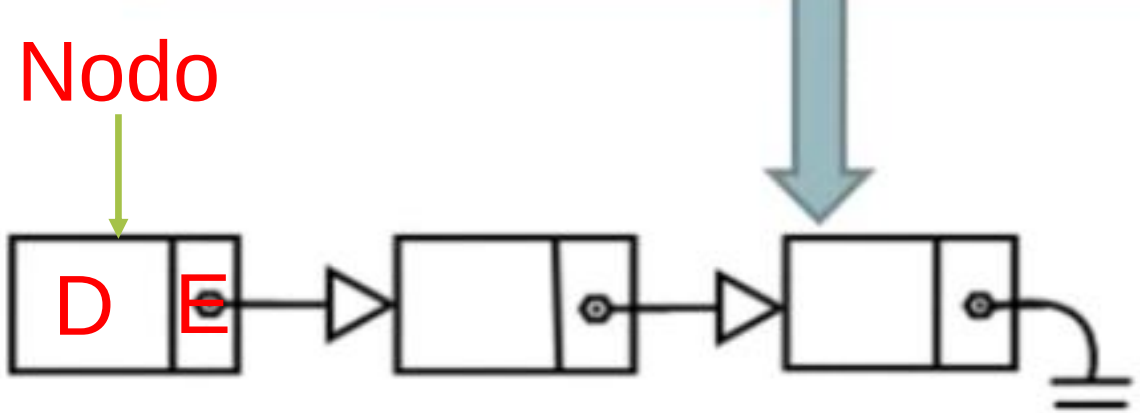
Estructuras de datos lineales



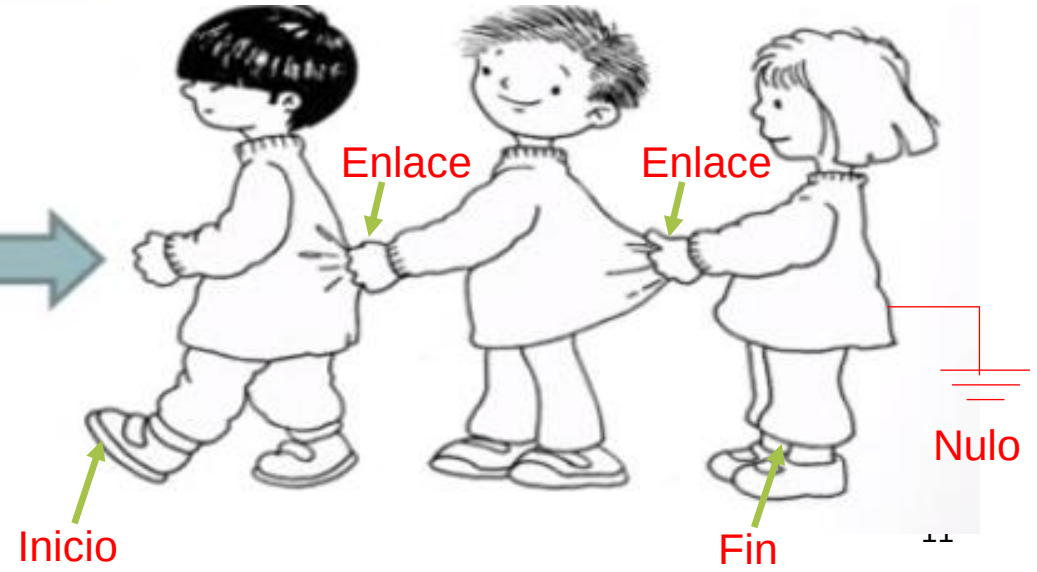
Listas Enlazadas

Una lista enlazada es una colección o secuencia de elementos dispuestos uno detrás de otro, en la que cada elemento se conecta al siguiente elemento por un "enlace" o "referencia".

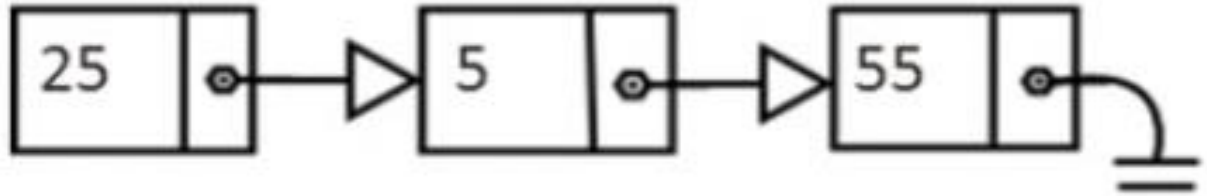
El nodo o elemento contiene el dato y un enlace al siguiente nodo o elemento



Lista enlazada



Comparación Listas Enlazadas vs Arreglo



Lista enlazada

Memoria Dinámica

Vector

25	5	55

Memoria Estática

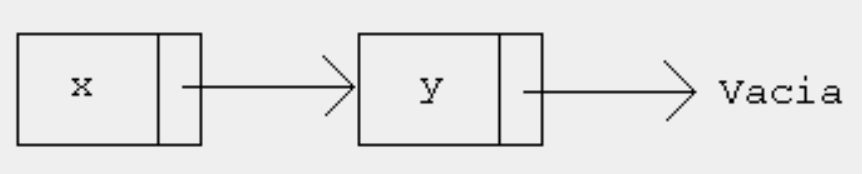
Listas Enlazadas

Una lista es una estructura de datos secuencial. En una lista enlazada: la posición del siguiente elemento de la estructura la determina el elemento actual. Es necesario almacenar al menos la posición de memoria del primer elemento. Además es dinámica, es decir, su tamaño cambia durante la ejecución del programa.

Una lista enlazada se puede definir recursivamente de la siguiente manera:

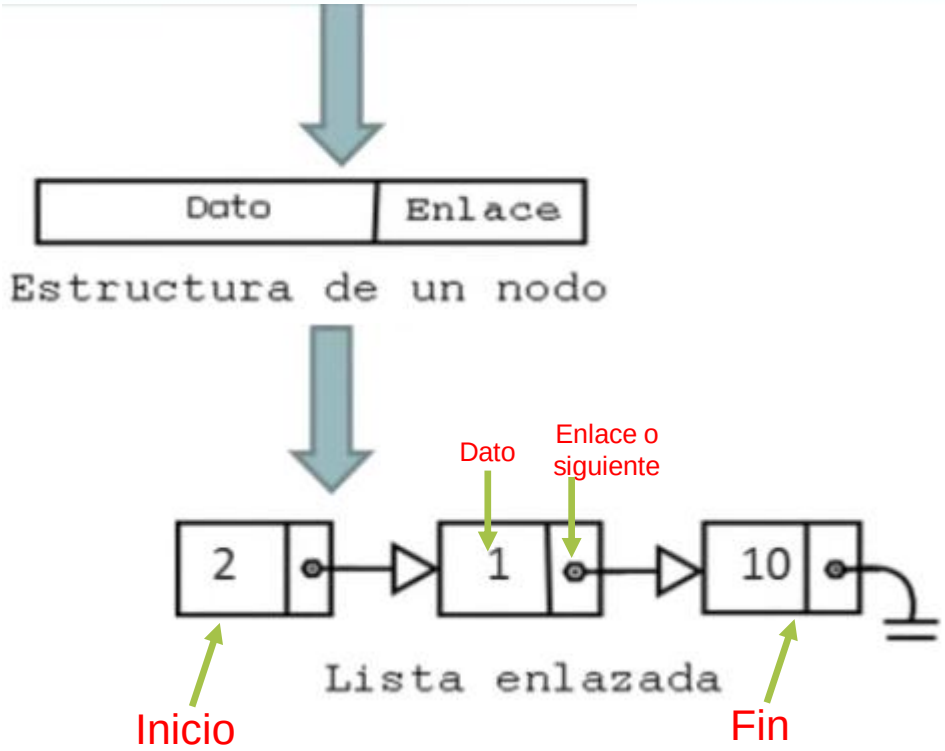
- Una lista enlazada es una estructura vacía o
- Un elemento de información y un enlace hacia una lista (un nodo).

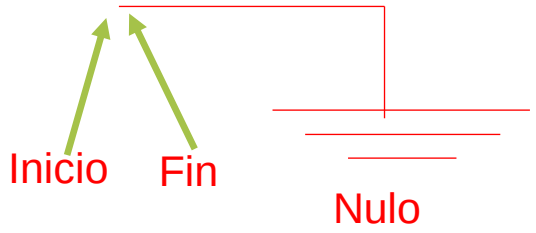
Gráficamente se suele representar así:



Estructura de una Lista Enlazada

- ✓ El Elemento principal es el "Nodo".
- ✓ El Nodo se compone de dos campos:
  1. La Información (Dato o Info)
  2. La Referencia (Enlace o Siguiente)

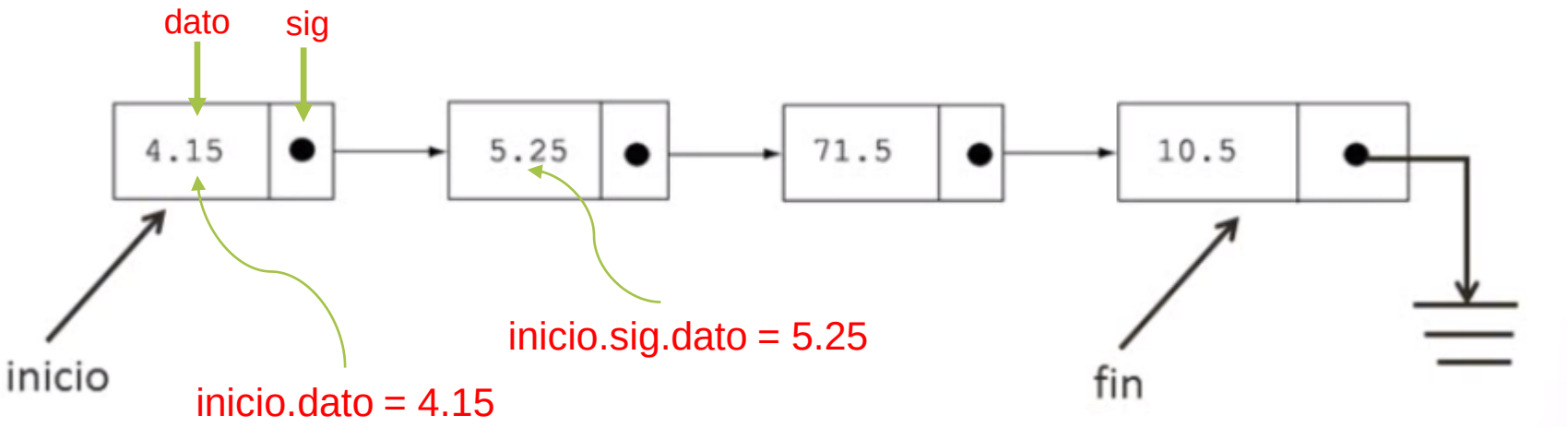




Si inicio o fin apuntan a nulo entonces la lista está vacía

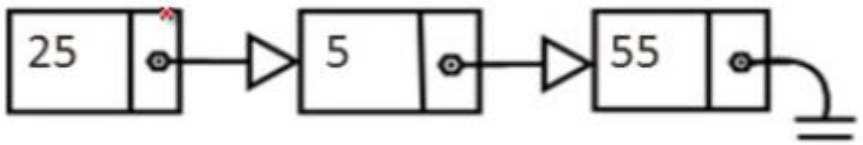
Operaciones en Listas Enlazadas

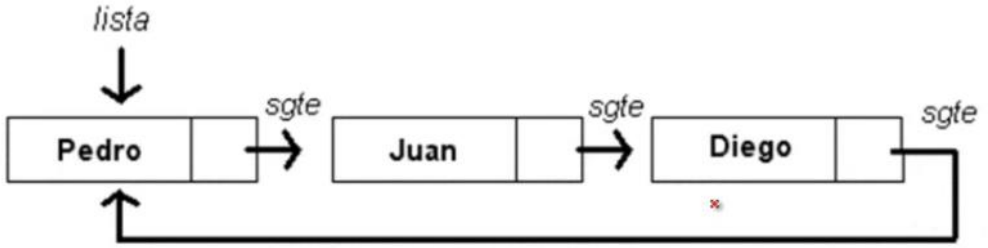
- ✓ Inicialización o Creación.
- ✓ Insertar elementos en la Lista.
- ✓ Eliminar elementos de la Lista.
- ✓ Buscar elementos en la Lista.
- ✓ Recorrer la Lista Enlazada.
- ✓ Comprobar si la Lista esta vacía.

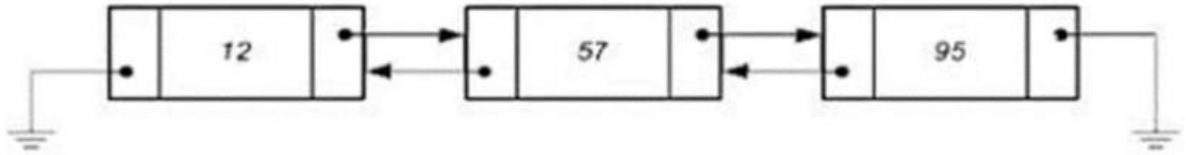


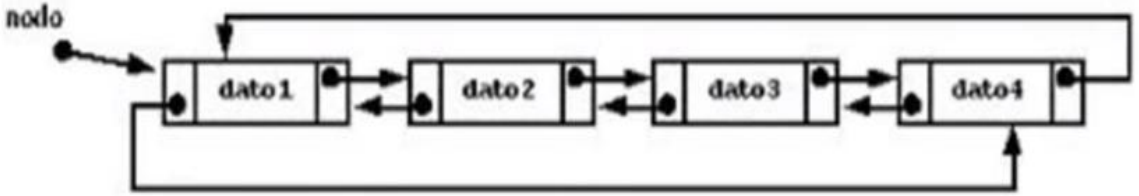
Clasificación de las Listas Enlazadas

- ✓ Listas Simplemente Enlazadas.
- ✓ Listas Doblemente Enlazadas.
- ✓ Lista Circular Simplemente Enlazada.
- ✓ Lista Circular Doblemente Enlazada.







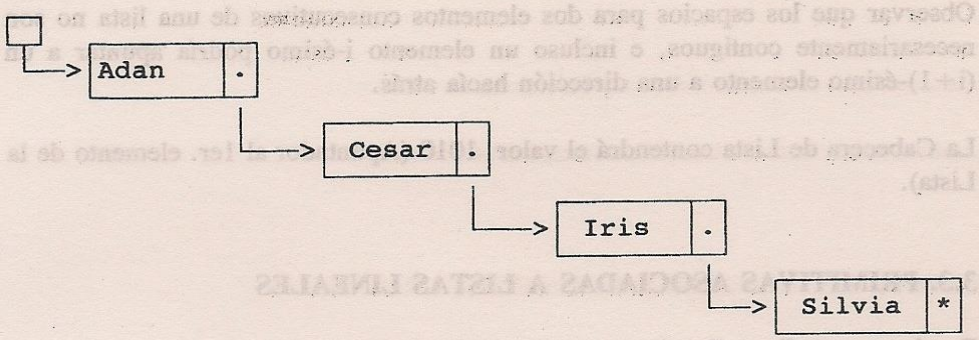


Clasificación de las Listas Enlazadas

Las listas se pueden dividir en 4 categorias:

CATEGORÍA	DESCRIPCIÓN
Simplemente Enlazadas	Cada nodo (elemento) contiene un único enlace que conecta ese nodo al nodo siguiente o nodo sucesor. La lista es eficiente en recorridos directos (“adelante”)
Doblemente enlazadas	Cada nodo contiene dos enlaces, uno a su nodo predecesor y el otro a su nodo sucesor. La lista es eficiente tanto como en recorrido directo (“adelante”) como en recorrido inverso (“atras”).
Circular simplemente enlazada	El ultimo elemento (cola) se enlaza al primer elemento (cabeza), de tal modo que la lista puede se recorrida de modo circular (“anillo”)
Circular doblemente enlazada	El ultimo elemento se enlaza con el primer elemento y viceversa. Esta lista se puede recorrer en modo circular tanto en direccion directa como viceversa.

Recordar....?



El esquema precedente puede estar implementado físicamente en un espacio de memoria cuya representación podría ser la siguiente:

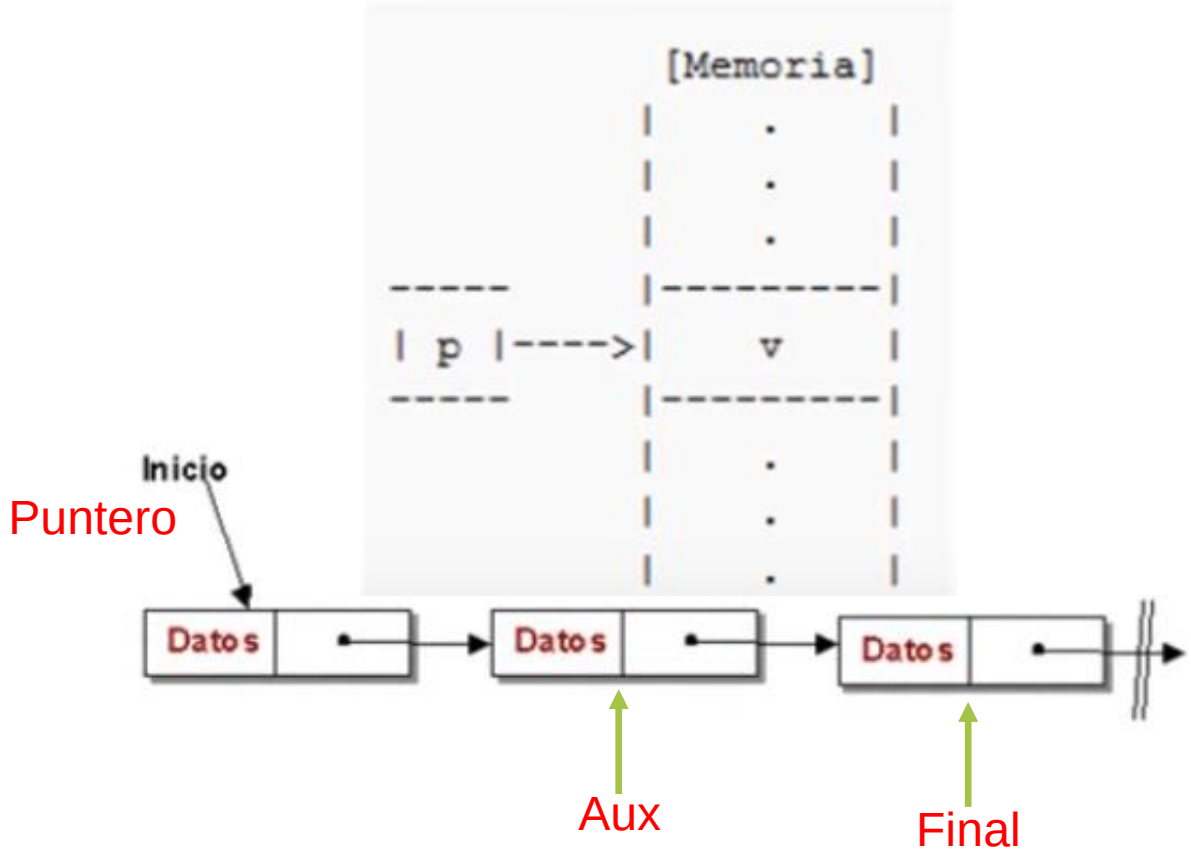
1010	Valores	Apunt.
1000		
1010	Adan	1030
1020		
1030	Cesar	1060
1040		
1050		
1060	Iris	1080
1070		
1080	Silvia	*
1090		
1100		

Nodo y Puntero

Un Nodo es un Elemento de una Lista Enlazada.
Cada nodo será una Estructura o Registro que dispondrá de varios Campos, y al menos uno de esos campos será un Puntero o referencia a otro nodo.

Estructura de un nodo

Un Puntero o Apuntador es una Variable que hace referencia a una región de la Memoria.



Los punteros hacen referencia a un Nodo por lo tanto son regularmente declarados de tipo Nodo

Clase Nodo

Estructura

Atributos del Nodo

```
public class Nodo {
    public int dato;
    public Nodo siguiente;//puntero enlace
//constructor para agregar un nodo al final
    public Nodo(int d) {
        this.dato = d;
        this.siguiente = null;
    }
//constructor para agregar un nodo al comienzo
    public Nodo(int d, Nodo n) {
        this.dato = d;
        this.siguiente = n;
    }
}
```

Puntero tipo Nodo

Constructor con sólo dato

Constructor con dato y puntero

Clase Nodo

```
public class Nodo {
    public int dato;
    public Nodo siguiente;//puntero enlace
//constructor para agregar un nodo al final
    public Nodo(int d) {
        this.dato = d;
        this.siguiente = null;
    }
//constructor para agregar un nodo al comienzo
    public Nodo(int d, Nodo n) {
        this.dato = d;
        this.siguiente = n;
    }
}
```

Nodo inicio = null;

Puntero

inicio

Nodo fin = null;

Puntero

fin

Nodo nuevo = new Nodo(4);

Puntero

nuevo

4

siguiente

Nulo

nuevo.siguiente = null;

inicio = nuevo;

inicio

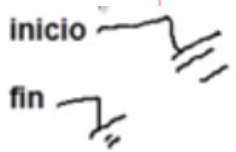
fin = nuevo;

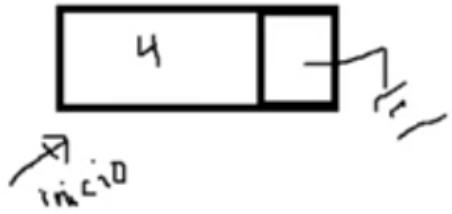
fin

Clase Lista v1

```java
public class Lista {
    protected Nodo inicio,fin; //punteros para saber donde están el inicio y el fin
    public Lista(){
        inicio=null;
        fin=null;
    }
    //metodo para agregar un nodo al inicio de la Lista
    public void agregarAlInicio(int elemento){
        //Crear el nodo
        inicio=new Nodo(elemento, inicio);
        if(fin==null){
            fin=inicio;
        }
    }
    //Método para mostrar los datos
    public void mostrarLista(){
        Nodo recorrer=inicio;
        System.out.println();
        while(recorrer!=null){
            System.out.print("["+recorrer.dato+"]--->");
            recorrer=recorrer.siguiente;
        }
    }
}
```

```java
public class Nodo {
    public int dato;
    public Nodo siguiente;//puntero enlace
//constructor para agregar un nodo al final
    public Nodo(int d) {
        this.dato = d;
        this.siguiente = null;
    }
//constructor para agregar un nodo al comienzo
    public Nodo(int d, Nodo n) {
        this.dato = d;
        this.siguiente = n;
    }
}
```





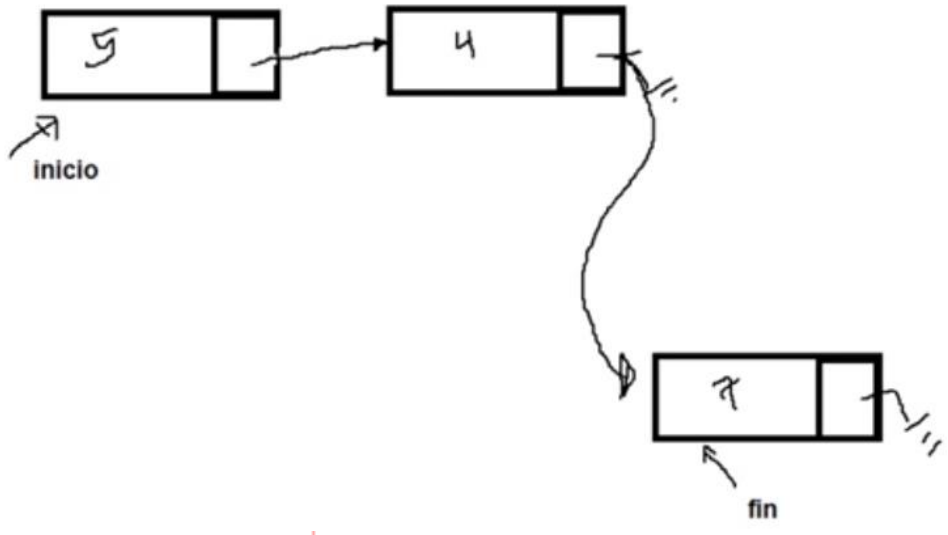
Uso de Lista v1

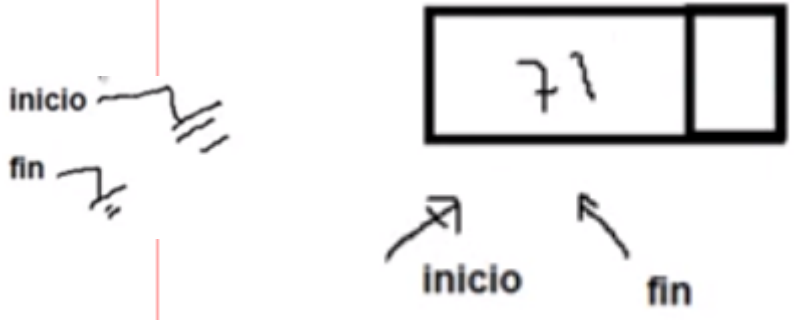
```
3   public class UsoLista {
4       public static void main(String[] args) {
5           Lista listal = new Lista();
6           int opcion = 0, el;
7           do {
8               try {
9                   opcion = Integer.parseInt(JOptionPane.showInputDialog(null,
10                          "1. Agregar un elemento al inicio de la lista\n"+
11                          "2. Mostrar datos de la lista\n"+
12                          "3. Salir","Menú de Opciones",3));
13                  switch (opcion) {
14                      case 1:
15                          try {
16                          el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
17                                  "Insertando al inicio", 3));
18                          //agregando al nodo
19                          listal.agregarAlInicio(el);
20                      } catch (NumberFormatException n) {
21                          JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
22                      }
23                      break;
24                      case 2:
25                          listal.mostrarLista();
26                          break;
27                      case 3:
28                          break;
29                      default:
30                          JOptionPane.showMessageDialog(null, "Opción incorrecta");
31                  }
32              } catch (Exception e) {
33                  JOptionPane.showMessageDialog(null, "Error " + e.getMessage());
34              }
35          } while (opcion != 3);
36      }
37  }
38
```

Clase Lista v2

```java
public class Lista {
    protected Nodo inicio,fin; //punteros para saber donde están el inicio y el fin
    public Lista(){
        inicio=null;
        fin=null;
    }
    //Metodo para saber si la lista está vacía
    public boolean estaVacia(){
        if(inicio==null){
            return true;
        }else{
            return false;
        }
    }
    //Método para insertar al final de la lista
    public void agregarAlFinal(int elemento){
        if(!estaVacia()){
            fin.siguiente=new Nodo(elemento);
            fin=fin.siguiente;
        }else{
            inicio=fin=new Nodo(elemento);
        }
    }
```

```java
public class Nodo {
    public int dato;
    public Nodo siguiente;//puntero enlace
//constructor para agregar un nodo al final
    public Nodo(int d) {
        this.dato = d;
        this.siguiente = null;
    }
//constructor para agregar un nodo al comienzo
    public Nodo(int d, Nodo n) {
        this.dato = d;
        this.siguiente = n;
    }
}
```





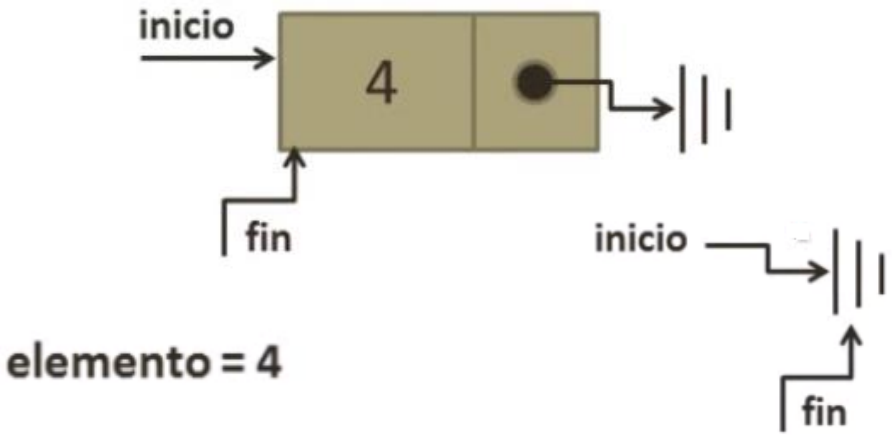
Uso de Lista v2

```
3   public class UsoLista {
4       public static void main(String[] args) {
5           Lista listal = new Lista();
6           int opcion = 0, el;
7           do {
8               try {
9                   opcion = Integer.parseInt(JOptionPane.showInputDialog(null,
10                          "1. Agregar un elemento al inicio de la lista\n"+
11                          "2. Agregar un elemento al final de la lista\n"+
12                          "3. Mostrar datos de la lista\n"+
13                          "4. Salir","Menú de Opciones",3));
14                  switch (opcion) {
15                      case 1:
16                          try {
17                          el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
18                                  "Insertando al inicio", 3));
19                          //agregando el nodo
20                          listal.agregarAlInicio(el);
21                          } catch (NumberFormatException n) {
22                              JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
23                          }
24                          break;
25                      case 2:
26                          try {
27                          el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
28                                  "Insertando al final", 3));
29                          //agregando el nodo
30                          listal.agregarAlFinal(el);
31                          } catch (NumberFormatException n) {
32                              JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
33                          }
34                              break;
35                      case 3:
36                          listal.mostrarLista();
37                          break;
38                      case 4:
39                          JOptionPane.showMessageDialog(null, "Programa Finalizado");
40                          break;
41                      default:
42                          JOptionPane.showMessageDialog(null, "Opción incorrecta");
43                  }
44              } catch (Exception e) {
45                  JOptionPane.showMessageDialog(null, "Error " + e.getMessage());
46              }
47          } while (opcion != 4);
48      }
49  }
```

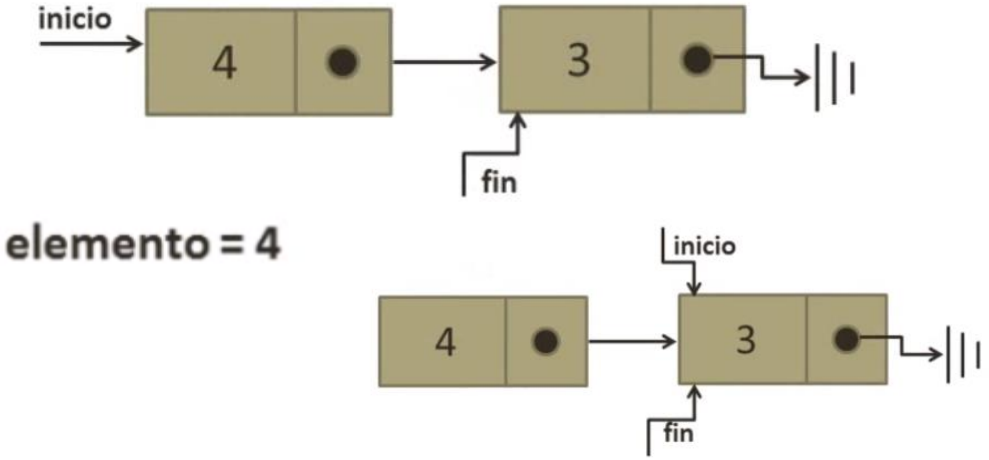
Eliminar un nodo del inicio

1. elemento = inicio de dato inicio.dato
2. Si inicio es igual a fin entonces
   1. Apuntar a inicio y fin a nulo
3. Si No entonces
   1. Apuntar inicio a inicio de siguiente inicio.siguiente
4. Retornar el valor de elemento

Eliminar un Nodo del Inicio
Si inicio es igual a fin



Eliminar un Nodo del Inicio
Si inicio y fin no son iguales



Clase Lista v3

```java
package Semana4;
public class Lista {
    protected Nodo inicio,fin; //punteros para saber donde están el inicio y el fin
    public Lista(){
        inicio=null;
        fin=null;
    }
    public int borrarDelInicio(){
        int elemento=inicio.dato;
        if(inicio==fin){
            inicio=null;
            fin=null;
        }else{
            inicio=inicio.siguiente;
        }
        return elemento;
    }
//Metodo para saber si la lista está vacía
    public boolean estaVacia(){
        if(inicio==null){
            return true;
        }else{
            return false;
        }
    }
    //Método para insertar al final de la lista
    public void agregarAlFinal(int elemento){
        if(!estaVacia()){
            fin.siguiente=new Nodo(elemento);
            fin=fin.siguiente;
        }else{
            inicio=fin=new Nodo(elemento);
        }
    }
    //metodo para agregar un nodo al inicio de la Lista
    public void agregarAlInicio(int elemento){
        //Crear el nodo
        inicio=new Nodo(elemento, inicio);
        if(fin==null){
            fin=inicio;
        }
    }
    //Método para mostrar los datos
    public void mostrarLista(){
        Nodo recorrer=inicio;
        System.out.println();
        while(recorrer!=null){
            System.out.print("["+recorrer.dato+"]--->");
            recorrer=recorrer.siguiente;
        }
    }
}
```

Uso de Lista v3

```java
package Semana4;
import javax.swing.JOptionPane;
public class UsoLista {
    public static void main(String[] args) {
        Lista listal = new Lista();
        int opcion = 0, el;
        do {
            try {
                opcion = Integer.parseInt(JOptionPane.showInputDialog(null,
                        "1. Agregar un elemento al inicio de la lista\n"+
                        "2. Agregar un elemento al final de la lista\n"+
                        "3. Mostrar datos de la lista\n"+
                        "4. Eliminar un elemento del inicio de la lista\n"+
                        "5. Salir","Menú de Opciones",3));
switch (opcion) {
    case 1:
        try {
        el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
                "Insertando al inicio", 3));
        //agregando el nodo
        listal.agregarAlInicio(el);
        } catch (NumberFormatException n) {
            JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
        }
        break;
    case 2:
        try {
                    el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
                            "Insertando al final", 3));
                    //agregando el nodo
                    listal.agregarAlFinal(el);
                } catch (NumberFormatException n) {
                    JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
                }
                    break;
                case 3:
                    listal.mostrarLista();
                    break;
                case 4:
                    el=listal.borrarDelInicio();
                    JOptionPane.showMessageDialog(null,"El elemento eliminado es: "
                            +el,"Eliminando nodo del inicio", JOptionPane.INFORMATION_MESSAGE);
                    break;
                case 5:
                    JOptionPane.showMessageDialog(null, "Programa Finalizado");
                    break;
                default:
                    JOptionPane.showMessageDialog(null, "Opción incorrecta");
                }
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, "Error " + e.getMessage());
            }
        } while (opcion != 5);
    }
}
```

Eliminar un nodo del final – Clase Lista v4

1. elemento = fin de dato — fin.dato
2. Si inicio es igual a fin entonces
   1. Apuntar a inicio y fin a nulo
3. Si No entonces
   1. Crear un Nodo temporal que apunte a inicio
   2. Mientras temporal de siguiente sea diferente de fin
      1. Apuntar temporal a temporal de siguiente — temporal.siguiente
   3. Apuntar fin a temporal
   4. Apuntar fin de siguiente a nulo. — fin.siguiente
4. Retornar el valor de elemento

```java
public class Lista {
    protected Nodo inicio, fin; //punteros para saber donde están el inicio y el fin
    public Lista() {
        inicio = null;
        fin = null;
    }
//Método para eliminar un Nodo del final
    public int borrarDelFinal() {
        int elemento = fin.dato;
        if (inicio == fin) {
            inicio = fin = null;
        } else {
            Nodo temporal = inicio;
            while (temporal.siguiente != fin) {
                temporal = temporal.siguiente;
            }
            fin = temporal;
            fin.siguiente = null;
        }
        return elemento;
    }
//Método para eliminar un Nodo del Inicio
    public int borrarDelInicio() {
        int elemento = inicio.dato;
        if (inicio == fin) {
            inicio = null;
            fin = null;
        } else {
            inicio = inicio.siguiente;
        }
        return elemento;
    }
//Metodo para saber si la lista está vacía
    public boolean estaVacia() {
        if (inicio == null) {
            return true;
        } else {
            return false;
        }
    }
//Método para insertar al final de la lista
    public void agregarAlFinal(int elemento) {
        if (!estaVacia()) {
            fin.siguiente = new Nodo(elemento);
            fin = fin.siguiente;
        } else {
            inicio = fin = new Nodo(elemento);
        }
    }
//metodo para agregar un nodo al inicio de la Lista
    public void agregarAlInicio(int elemento) {
        //Crear el nodo
        inicio = new Nodo(elemento, inicio);
        if (fin == null) {
            fin = inicio;
        }
    }
//Método para mostrar los datos
    public void mostrarLista() {
        Nodo recorrer = inicio;
        System.out.println();
        while (recorrer != null) {
            System.out.print("[" + recorrer.dato + "]--->");
            recorrer = recorrer.siguiente;
        }
```

Uso de Lista v4

```
3   public class UsoLista {
4       public static void main(String[] args) {
5           Lista listal = new Lista();
6           int opcion = 0, el;
7           do {
8               try {
9                   opcion = Integer.parseInt(JOptionPane.showInputDialog(null,
10                          "1. Agregar un elemento al inicio de la lista\n"+
11                          "2. Agregar un elemento al final de la lista\n"+
12                          "3. Mostrar datos de la lista\n"+
13                          "4. Eliminar un elemento del inicio de la lista\n"+
14                          "5. Eliminar un elemento del final de la lista\n"+
15                          "6. Salir","Menú de Opciones",3));
16                  switch (opcion) {
17                      case 1:
18                          try {
19                          el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
20                                  "Insertando al inicio", 3));
21                          //agregando el nodo
22                          listal.agregarAlInicio(el);
23                      } catch (NumberFormatException n) {
24                          JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
25                      }
26                      break;
27                      case 2:
28                          try {
29                          el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
30                                  "Insertando al final", 3));
31                          //agregando el nodo
```

```
                        tal.agregarAlFinal(el);
                        (NumberFormatException n) {
                        tionPane.showMessageDialog(null, "Error " + n.getMessage());

                        ak;

                        tal.mostrarLista();
                        ak;

                        listal.borrarDelInicio();
                        tionPane.showMessageDialog(null,"El elemento eliminado es: "
                            +el,"Eliminando nodo del inicio", JOptionPane.INFORMATION_MESSAGE);
                        ak;

                        listal.borrarDelFinal();
                        tionPane.showMessageDialog(null,"El elemento eliminado es: "
                            +el,"Eliminando nodo del final", JOptionPane.INFORMATION_MESSAGE);

                        ak;
51                      case 6:
52                          JOptionPane.showMessageDialog(null, "Programa Finalizado");
53                          break;
54                      default:
55                          JOptionPane.showMessageDialog(null, "Opción incorrecta");
56                  }
58              } catch (Exception e) {
58                  JOptionPane.showMessageDialog(null, "Error " + e.getMessage());
59              }
60          } while (opcion != 6);
61      }
```

Otras operaciones sobre las Listas simplemente enlazadas

Buscar un Elemento

1. Crear un Nodo temporal y apuntarlo a inicio
2. Mientras temporal sea diferente de nulo Y temporal de dato seas diferente de elemento Hacer
   1. Apuntar temporal a temporal de siguiente
3. Retornar temporal es diferente de nulo

```
92      //Metodo para buscar un elemento
93      public boolean estaEnLaLista(int elemento){
94          Nodo temporal = inicio;
95          while(temporal!=null && temporal.dato!=elemento){
96              temporal=temporal.siguiente;
97          }
98          return temporal!=null;
99      }
```

Otras operaciones sobre las Listas simplemente enlazadas

Eliminar un Nodo en Especifico

Si la Lista No esta vacía entonces
1. Si inicio es igual a fin Y elemento igual a inicio de dato Entonces
   1. Apuntar inicio y fin a nulo
2. Si No Si elemento igual a inicio de dato Entonces
   1. Apuntar inicio a inicio de siguiente
3. Si No
   1. Crear dos Nodos, anterior y temporal
   2. Apuntar anterior a inicio
   3. Apuntar temporal a inicio de siguiente
   4. Mientras temporal sea diferente de nulo y temporal de dato diferente de elemento Hacer
      1. Apuntar anterior a anterior de siguiente
      2. Apuntar temporal a temporal de siguiente
   5. Si temporal es diferente de nulo Entonces
      1. Apuntar anterior de siguiente a temporal de siguiente
      2. Si temporal es igual a fin Entonces
         1. Apuntar fin a anterior

```
68      //Metodo para eliminar un nodo en especifico
69      public void eliminar(int elemento){
70          if(!estaVacia()){
71              if(inicio==fin && elemento==inicio.dato){
72                  inicio=fin=null;
73              }else if(elemento==inicio.dato){
74                  inicio=inicio.siguiente;
75              }else{
76                  Nodo anterior,temporal;
77                  anterior=inicio;
78                  temporal=inicio.siguiente;
79                  while(temporal!=null && temporal.dato!=elemento){
80                      anterior=anterior.siguiente;
81                      temporal=temporal.siguiente;
82                  }
83                  if(temporal!=null){
84                      anterior.siguiente=temporal.siguiente;
85                      if(temporal==fin){
86                          fin=anterior;
87                      }
88                  }
89              }
90          }
91      }
```

Eliminar un nodo del final – Clase Lista v5

```java
package Semana4;
public class Lista {
    protected Nodo inicio, fin; //punteros para saber donde están el inicio y el fin
    public Lista() {
        inicio = null;
        fin = null;
    }
//Método para eliminar un Nodo del final
    public int borrarDelFinal() {
        int elemento = fin.dato;
        if (inicio == fin) {
            inicio = fin = null;
        } else {
            Nodo temporal = inicio;
            while (temporal.siguiente != fin) {
                temporal = temporal.siguiente;
            }
            fin = temporal;
            fin.siguiente = null;
        }
        return elemento;
    }
    //Método para eliminar un Nodo del Inicio
    public int borrarDelInicio() {
        int elemento = inicio.dato;
        if (inicio == fin) {
            inicio = null;
            fin = null;
        } else {
            inicio = inicio.siguiente;
        }
        return elemento;
    }
//Metodo para saber si la lista está vacía
    public boolean estaVacia() {
        if (inicio == null) {
            return true;
        } else {
            return false;
        }
    }
    //Método para insertar al final de la lista
    public void agregarAlFinal(int elemento) {
        if (!estaVacia()) {
            fin.siguiente = new Nodo(elemento);
            fin = fin.siguiente;
        } else {
            inicio = fin = new Nodo(elemento);
        }
    }
    //metodo para agregar un nodo al inicio de la Lista
    public void agregarAlInicio(int elemento) {
        //Crear el nodo
        inicio = new Nodo(elemento, inicio);
        if (fin == null) {
            fin = inicio;
        }
    }
    //Método para mostrar los datos
    public void mostrarLista() {
        Nodo recorrer = inicio;
        System.out.println();
        while (recorrer != null) {
            System.out.print("[" + recorrer.dato + "]--->");
            recorrer = recorrer.siguiente;
        }
    }
```

Eliminar un nodo del final – Clase Lista v5

```
68      //Metodo para eliminar un nodo en especifico
69      public void eliminar(int elemento){
70          if(!estaVacia()){
71              if(inicio==fin && elemento==inicio.dato){
72                  inicio=fin=null;
73              }else if(elemento==inicio.dato){
74                  inicio=inicio.siguiente;
75              }else{
76                  Nodo anterior,temporal;
77                  anterior=inicio;
78                  temporal=inicio.siguiente;
79                  while(temporal!=null && temporal.dato!=elemento){
80                      anterior=anterior.siguiente;
81                      temporal=temporal.siguiente;
82                  }
83                  if(temporal!=null){
84                      anterior.siguiente=temporal.siguiente;
85                      if(temporal==fin){
86                          fin=anterior;
87                      }
88                  }
89              }
90          }
91      }
```

```
92      //Metodo para buscar un elemento
93      public boolean estaEnLaLista(int elemento){
94          Nodo temporal = inicio;
95          while(temporal!=null && temporal.dato!=elemento){
96              temporal=temporal.siguiente;
97          }
98          return temporal!=null;
99      }
```

Uso de Lista v5

```java
package Semana4;
import javax.swing.JOptionPane;
public class UsoLista {
    public static void main(String[] args) {
        Lista listal = new Lista();
        int opcion = 0, el;
        do {
            try {
                opcion = Integer.parseInt(JOptionPane.showInputDialog(null,
                        "1. Agregar un elemento al inicio de la lista\n"+
                        "2. Agregar un elemento al final de la lista\n"+
                        "3. Mostrar datos de la lista\n"+
                        "4. Eliminar un elemento del inicio de la lista\n"+
                        "5. Eliminar un eleme
                        "6. Eliminar un eleme
                        "7. Buscar un Element
                        "8. Salir","Menú de O
                switch (opcion) {
                    case 1:
                        try {
                        el = Integer.parseInt
                                "Insertando a
                        //agregando el nodo
                        listal.agregarAlInici
                    } catch (NumberFormatExce
                        JOptionPane.showMessa
                    }
                    break;
                    case 2:
                        try {
                        el = Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el elemento",
                                "Insertando al final", 3));
                        //agregando el nodo
                        listal.agregarAlFinal(el);
                    } catch (NumberFormatException n) {
                        JOptionPane.showMessageDialog(null, "Error " + n.getMessage());
                    }
                        break;
                    case 3:
                        listal.mostrarLista();
                        break;
                    case 4:
                        el=listal.borrarDelInicio();
                        JOptionPane.showMessageDialog(null,"El elemento eliminado es: "
                                +el,"Eliminando nodo del inicio", JOptionPane.INFORMATION_MESSAGE);
                        break;
                    case 5:
                        el=listal.borrarDelFinal();
                        JOptionPane.showMessageDialog(null,"El elemento eliminado es: "
                                +el,"Eliminando nodo del final", JOptionPane.INFORMATION_MESSAGE);
                        break;
```

Uso de Lista v5

```
52                  case 6:
53                      el=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el "+
54                              "elemento a eliminar...", "Eliminando nodos en especifico"
55                              ,JOptionPane.INFORMATION_MESSAGE));
56                      if(listal.estaEnLaLista(el)){
57                      listal.eliminar(el);
58                      JOptionPane.showMessageDialog(null,"El elemento eliminado es: "
59                              +el,"Eliminando nodos en especifico", JOptionPane.INFORMATION_MESSAGE);
60                      }else{
61                      JOptionPane.showMessageDialog(null, "El elemento "+el+" no está en la lista",
62                              "Nodo no encontrado",JOptionPane.INFORMATION_MESSAGE);
63                      }
64                      break;
65                  case 7:
66                      el=Integer.parseInt(JOptionPane.showInputDialog(null,"Ingresa el "+
67                              "elemento a buscar...", "Buscando nodos en la lista"
68                              ,JOptionPane.INFORMATION_MESSAGE));
69                      if(listal.estaEnLaLista(el)){
70                      JOptionPane.showMessageDialog(null, "El elemento "+el+" Si está en la lista",
71                              "Nodo encontrado",JOptionPane.INFORMATION_MESSAGE);
72                      }else{
73                      JOptionPane.showMessageDialog(null, "El elemento "+el+" no está en la lista",
74                              "Nodo no encontrado",JOptionPane.INFORMATION_MESSAGE);
75                      }
76                      break;
```

```
77                      case 8:
78                          JOptionPane.showMessageDialog(null, "Programa Finalizado");
79                          break;
80                      default:
81                          JOptionPane.showMessageDialog(null, "Opción incorrecta");
82                  }
                } catch (Exception e) {
84                  JOptionPane.showMessageDialog(null, "Error " + e.getMessage());
85              }
86          } while (opcion != 8);
87      }
88  }
```

Universidad Nacional Mayor de San Marcos
Universidad del Perú. Decana de América

Guía de Laboratorio

Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América