UNIVERSIDAD NACIONAL DE CAJAMARCA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS

ALGORITMOS Y ESTRUCTURA DE DATOS II
02: LISTAS ENLAZADAS SIMPLES

INTRODUCCIÓN

Semana 02 - Listas enlazadas simples

REALICE LA SIGUIENTE TAREA

- Imagine que se tiene una lista que tiene por ahora un elemento, añada el resto de elementos **AL INICIO**, en el orden indicado y muestre su resultado final.

https://visualgo.net/es/list

RECORDEMOS

- ¿Qué es un nodo?
- ¿Para qué sirve un nodo?
- ¿Cuál es el código necesario para implementar un nodo en java?
- En el ejemplo de la diapositiva anterior identifique los nodos existentes.

QUÉ PASARÍA SI…

- … me piden que implemente una colección con características específicas, es decir una serie de métodos personalizados, ¿Cómo lo implementaría?

LOGRO ESPERADO

- Al término de la sesión, el estudiante diseña programa Java que permita realizar las tareas básicas de una colección, verificando el buen funcionamiento del mismo y aplicando su propuesta para la solución de un problema real.

DESARROLLO DEL TEMA

Estructura de Datos

LISTAS ENLAZADAS

- La forma más simple de estructura dinámica es la lista enlazada. En esta forma los nodos se organizan de modo que cada uno referencia al siguiente, y el último no hace referencia a nada, es decir, la referencia del nodo siguiente vale NULL.

- En las listas enlazadas existe un nodo especial: el primero.
- Por lo general, un programa accede a una lista enlazada mediante una referencia al primer nodo en la lista, en el caso de la figura se llama “primero”. Es llamado también inicio, lista, cabeza, raíz, entre otros.
- Cuando la referencia que usamos para acceder a la lista vale NULL, diremos que la lista está vacía.

LISTA ENLAZADA – DECLARACIONES DE TIPOS

- Como ya se vio anteriormente usaremos la declaración de la siguiente forma:
- ¿Considera pueda tener sentido los getter y setters así como también el toString()?

```
public class Nodo {
    int dato;
    Nodo sgt;

    public Nodo(int dato) {
        this.dato = dato;
        this.sgt = null;
    }
    public Nodo(int dato, Nodo sgt) {
        this.dato = dato;
        this.sgt = sgt;
    }
}
```

LISTA ENLAZADA - DECLARACIONES DE TIPOS

- Para construir la lista se hace necesario crear la clase ListaEnlazada, que estará formada de nodos.
- La referencia primero (también se puede llamar cabeza) se ha inicializado en el constructor a un valor nulo, es decir, a lista vacía.

```
public class ListaEnlazada {
    Nodo primero;

    public ListaEnlazada() {
        primero = null;
    }
}
```

OPERACIONES BÁSICAS CON LISTAS

- Entre las operaciones básicas a realizar se tiene:
  - Añadir o insertar.
  - Buscar o localizar.
  - Borrar.
  - Moverse a través de una lista, anterior, siguiente, primero.
- Cada una de estas operaciones tendrá varios casos especiales, por ejemplo, no será lo mismo insertar un nodo en una lista vacía, o al principio de una lista no vacía, o la final, o en una posición intermedia.

https://visualgo.net/es/list

INSERTAR NODOS EN UNA LISTA ENLAZADA

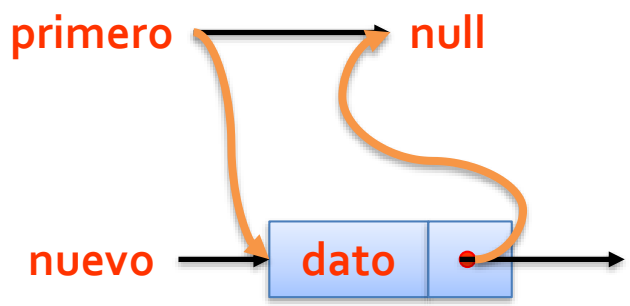
- **Insertar un elemento en una lista vacía al inicio:**
  - Este es, evidentemente, el caso más sencillo.
  - Partiremos de que ya tenemos una referencia a “primero” que valdrá NULL (lista está vacía),
  - Además el nodo a insertar y, por supuesto una referencia que apunte a él,


```
primero = null;
//constructor de la lista
```


```
Nodo nuevo = new Nodo();
```

INSERTAR NODOS EN UNA LISTA ENLAZADA

- **Insertar un elemento en una lista vacía al inicio:**
  - El proceso es muy simple, bastará con que:
  - nuevo.sgt apunte a primero.
  - Primero referencia a nuevo.

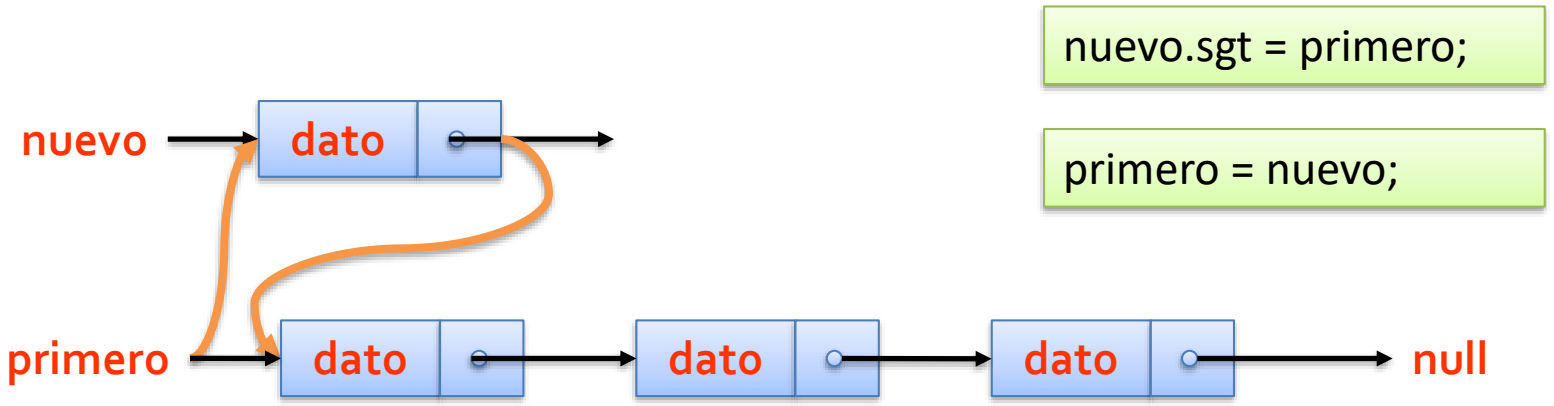



```
nuevo.sgt = primero;
```

```
primero = nuevo;
```

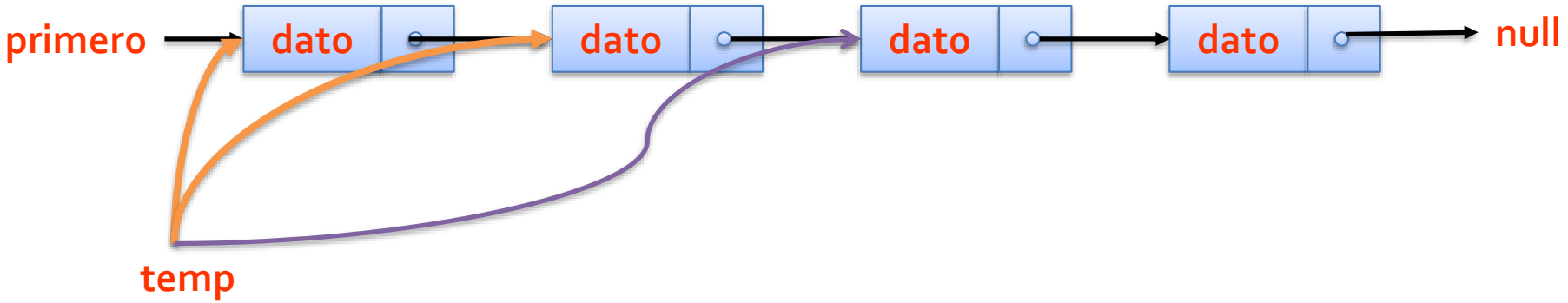
INSERTAR NODOS EN UNA LISTA ENLAZADA*

- **Insertar un elemento en la primera posición:**
  - El proceso es muy simple y bastante similar:
  - nuevo.sgt apunte a primero.
  - Primero referencia a nuevo.



MOSTRAR LOS ELEMENTOS DE LA LISTA

- Para mostrar los elementos de una lista se necesita:
  - Del tipo nodo se crea un temp que se iguala a primero
  - Mientras temp sea diferente de null
    - Mostrar temp.dato
    - Actualizar temp a temp.sgt



temp = primero;

Muestra temp.dato;

temp = temp.sgt;

EJEMPLO DE UNA LISTA ENLAZADA SIMPLE

- **Clase Nodo**

```
public class Nodo {
    int dato;
    Nodo sgt;

    public Nodo(int dato, Nodo sgt) {
        this.dato = dato;
        this.sgt = sgt;
    }

    public Nodo(int dato) {
        this(dato,null);
    }
}
```

- **Clase Lista Enlazada**

```
public class ListaEnlazada {
     Nodo primero;
     public ListaEnlazada() {
          primero = null;
     }
     public void insertaIni(int n) {
          Nodo nuevo = new Nodo(n);
          nuevo.sgt = primero;
          primero = nuevo;
     }
     public void imprimeLis() {
          Nodo temp = primero;
          while (temp!=null) {
               System.out.print(temp.dato + "\t");
               temp = temp.sgt;
          }
     }
}
```

EJEMPLO DE UNA LISTA ENLAZADA SIMPLE

- **Clase de prueba**

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("numero de elementos: ");
    int n = sc.nextInt();

    ListaEnlazada listita = new ListaEnlazada();
    for (int i= 1; i<=n;i++){
        System.out.print("dato " + i + ": ");
        int temp = sc.nextInt();
        listita.insertaIni(temp);
    }
    listita.imprimeLis();
}
```

FUNCIONES TIPO PARA EL MANEJO DE LISTAS ENLAZADAS

Inserta un nodo al inicio

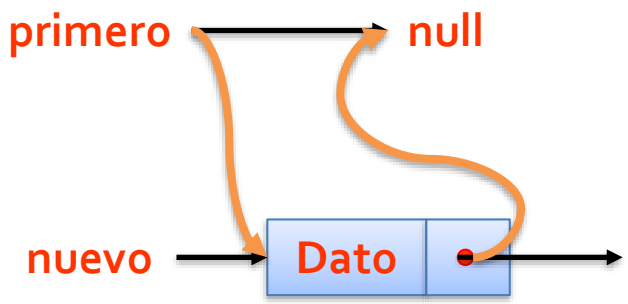
```
public void insertaIni(int n) {
    Nodo nuevo = new Nodo(n);
    nuevo.sgt = primero;
    primero = nuevo;
}
```

Mostrar una lista

```
public void imprimeLis() {
    Nodo temp = primero;
    while (temp!=null) {
        System.out.print(temp.dato +
"\t");
        temp = temp.sgt;
    }
}
```

INSERTAR NODOS EN UNA LISTA ENLAZADA

- **Insertar un elemento en una lista vacía en la última posición:**
  - El proceso es muy simple:
  - nuevo.sgt apunte a null.
  - primero apunte a nuevo.



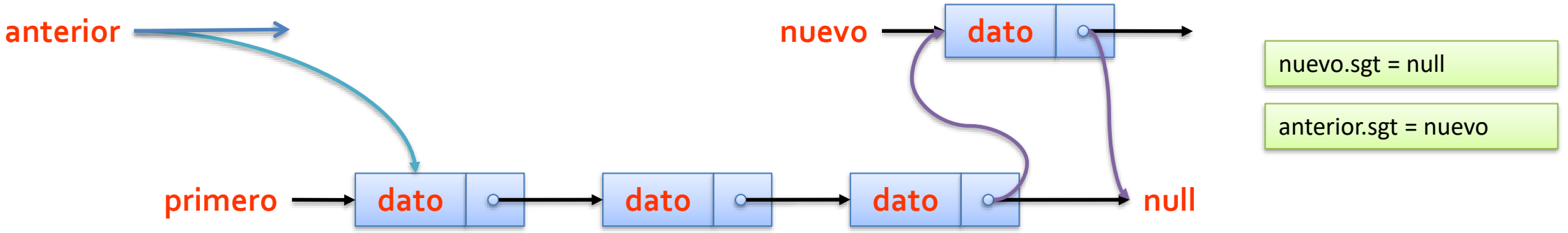

```
nuevo.sgt = null;
```

```
primero = nuevo;
```

INSERTAR NODOS EN UNA LISTA ENLAZADA*

- **Insertar un elemento en la última posición:**
  - Partimos de una lista no vacía:
  - Necesitamos una referencia que señale al último elemento de la lista (anterior).
  - La manera de conseguirlo es empezar por el primero y avanzar hasta que el nodo tenga como siguiente el valor null.
  - Hacer que nuevo.sgt sea null.
  - Hacer que anterior.sgt sea nuevo.

```
anterior = primero
while(anterior.sgt != null)
    anterior=anterior.sgt
```



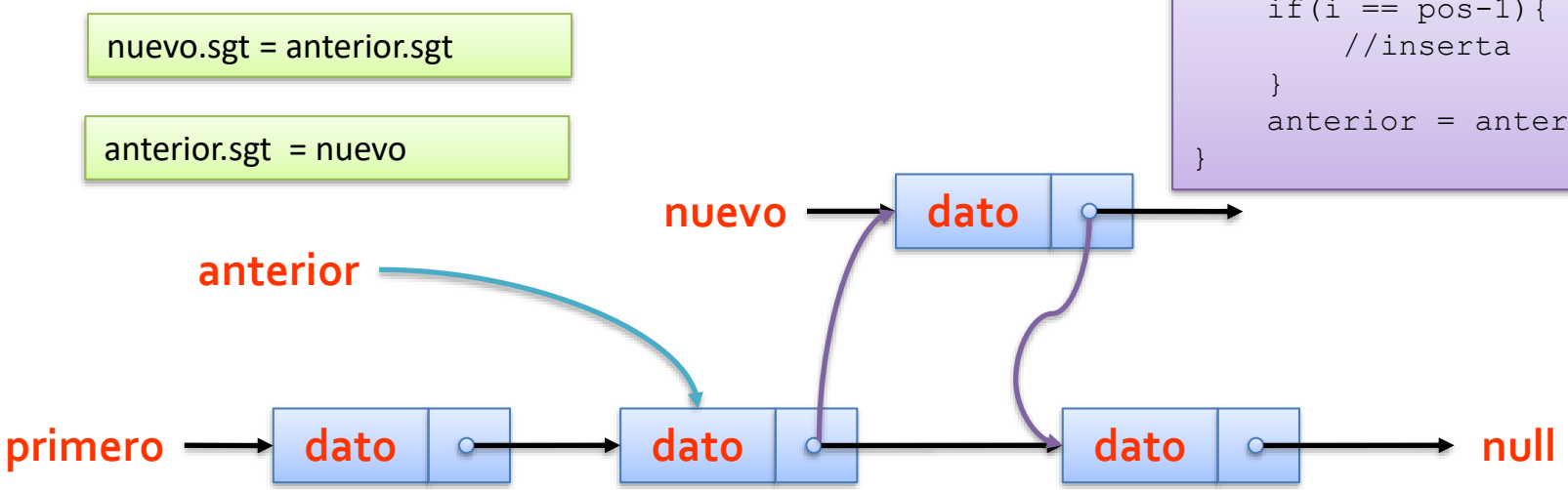
INSERTAR NODOS EN UNA LISTA ENLAZADA*

- **Insertar nodo a continuación de un nodo cualquiera:**
  - Podemos considerar el caso anterior como un caso particular de este. Ahora el nodo "anterior" será aquel a continuación del cual insertaremos el nuevo nodo :
  - Suponemos que ya disponemos del nuevo nodo a insertar, apuntado por nuevo, y un puntero al nodo a continuación del que lo insertaremos (anterior). El proceso a seguir será.
  - Hacer que nuevo.sgt señale a anterior.sgt
  - Hacer que anterior.sgt señale a nuevo.

```
anterior = primero;
for(int i=1;anterior!= null;i++){
    if(i == pos-1){
        //inserta
    }
    anterior = anterior.sgt;
}
```

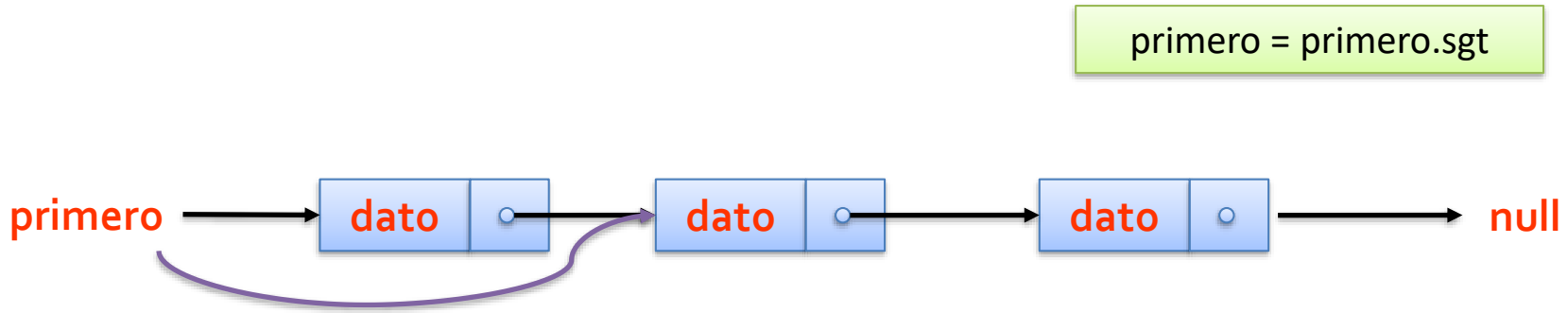
nuevo.sgt = anterior.sgt

anterior.sgt = nuevo



ELIMINAR ELEMENTOS DE UNA LISTA*

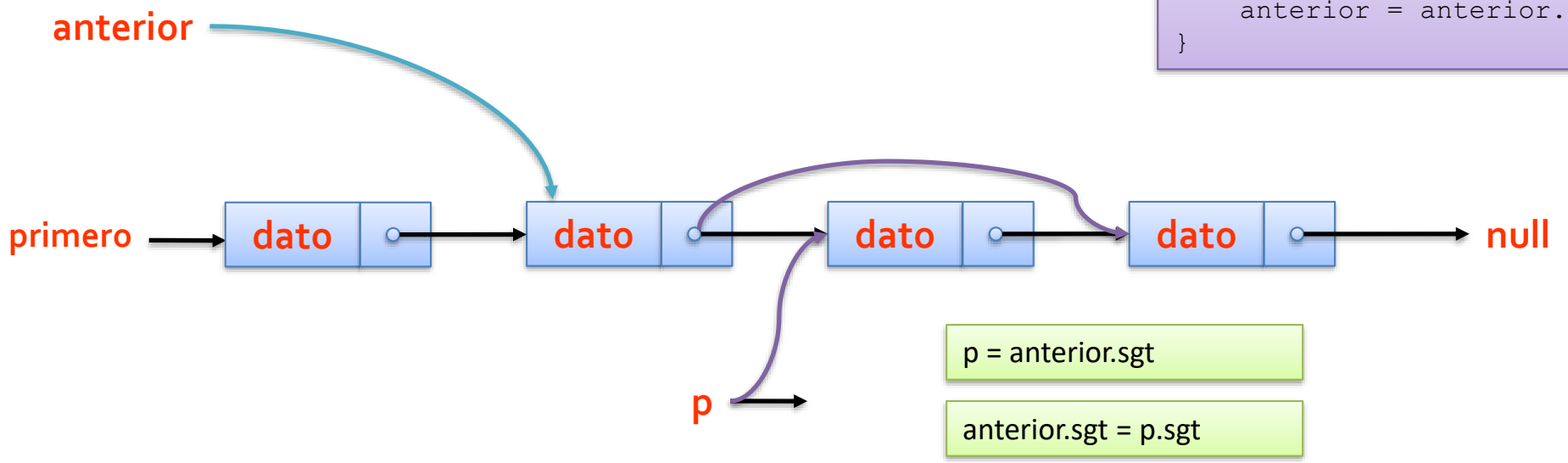
- **Eliminar el primer nodo de la lista:**
  - Asumimos que tenemos una lista (primero) con uno o más nodos.
  - primero apunta al siguiente nodo (segundo nodo)
  - Java libera la memoria asignada al elemento a eliminar



ELIMINAR ELEMENTOS DE UNA LISTA *

- **Elimina un nodo cualquiera de una lista (alternativa 1).**
  - Partimos de una lista no vacía con al menos dos elementos:
  - Necesitamos además una referencia que señale al nodo anterior al que queremos eliminar (anterior). Hacemos que p apunte al nodo que queremos borrar
  - Luego indicamos que anterior.sgt apunte hacia p.sgt
  - Java libera la memoria del elemento eliminado.

```
anterior = primero;
for(int i=1;anterior.sgt!=null;i++){
    if(i == pos-1){
        //elimina
    }
    anterior = anterior.sgt;
}
```

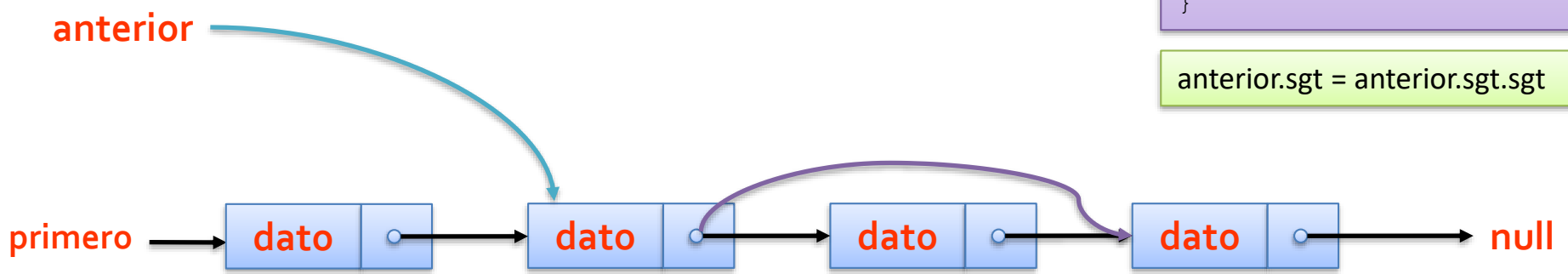


ELIMINAR ELEMENTOS DE UNA LISTA *

- **Elimina un nodo cualquiera de una lista (alternativa 2).**
  - Partimos de una lista no vacía con al menos dos elementos:
  - Necesitamos además una referencia que señale al nodo anterior al que queremos eliminar (anterior).
  - Luego indicamos que anterior.sgt apunte hacia anterior.sgt.sgt
  - Java libera la memoria del elemento eliminado.

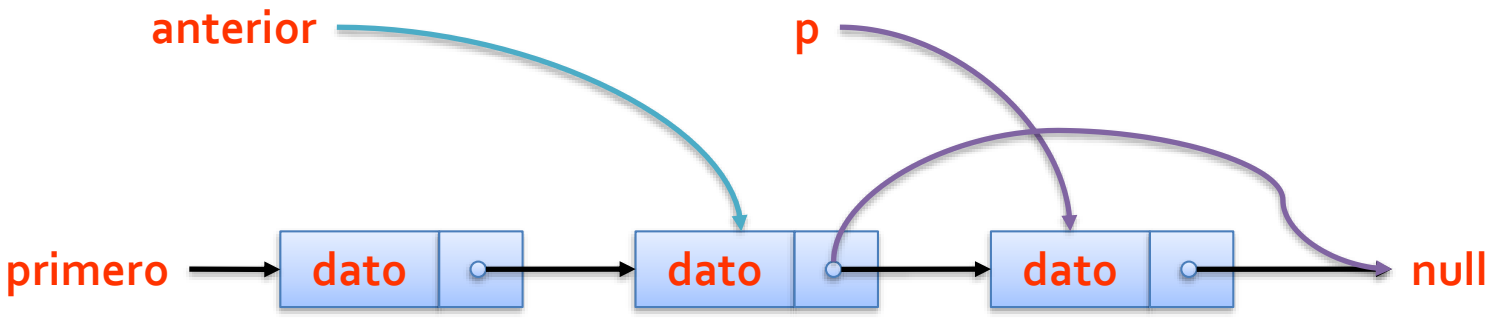
```
anterior = primero;
for(int i=1;anterior.sgt!=null;i++){
    if(i == pos-1){
        //elimina
     return();
    }
    anterior = anterior.sgt;
}
```

anterior.sgt = anterior.sgt.sgt



ELIMINAR ELEMENTOS DE UNA LISTA *

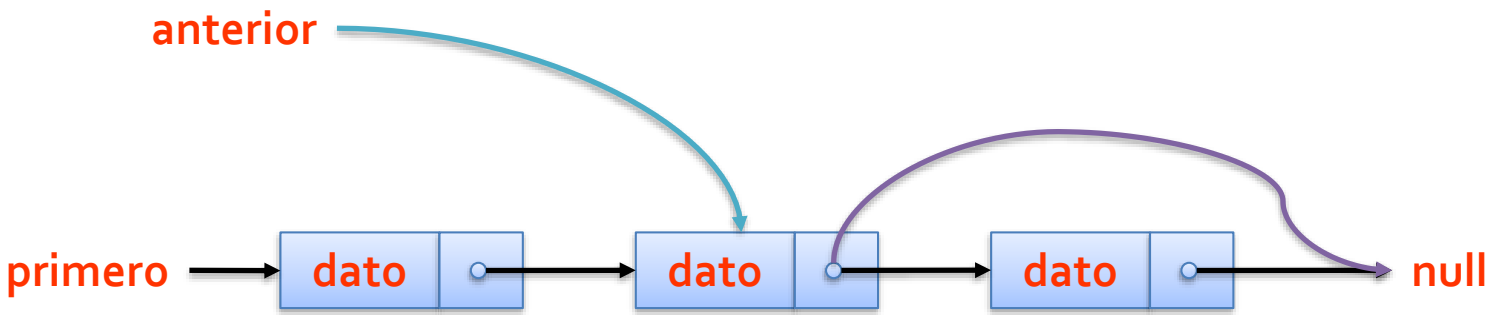
- **Eliminar el último elemento de la lista (alternativa 1):**
  - En nuestro ejemplo partimos de una lista no vacía:
  - Necesitamos un puntero que señale al penúltimo elemento de la lista (anterior) y que p apunte al elemento a eliminar (último).
  - Luego indicamos que anterior.sgt apunte hacia p.sgt
  - El garbage collector libera la memoria del elemento a eliminar.




```
p = primero
while(p.sgt != null){
    anterior = p
    p = p.sgt
}
anterior.sgt = p.sgt
```

ELIMINAR ELEMENTOS DE UNA LISTA *

- **Eliminar el último elemento de la lista (alternativa 2):**
  - En nuestro ejemplo partimos de una lista no vacía:
  - Necesitamos un puntero que señale al penúltimo elemento de la lista (anterior).
  - Luego indicamos que anterior.sgt apunte hacia null
  - El garbage collector libera la memoria del elemento a eliminar.




```
anterior = primero
while(anterior.sgt.sgt != null){
      anterior = anterior.sgt
}
anterior.sgt = null
```

ELIMINAR ELEMENTOS DE UNA LISTA

- Elimina toda una lista
- Hacemos que primero apunte a null.


```
primero = null
```

EVALUACIÓN DEL TEMA DESARROLLADO

Reflexionemos!

REFLEXIONEMOS SOBRE LO APRENDIDO

- ¿Qué parte del código modificaría para que la lista enlazada estudiada permita registrar los datos de una persona?
- Describa el algoritmo o pasos necesarios para insertar un elemento al inicio, al final y en un posición cualquiera. Hacerlo a través de un gráfico.
- ¿Qué dificultades o desventajas encuentra en esta lista enlazada? ¿Podría mejorarla?

EJERCICIOS DE APLICACIÓN

EJERCICIOS DE APLICACIÓN

Agregar un método a la lista enlazada que:

- Determine la cantidad de elementos que tiene la lista.
- Ordene los elementos de la lista
- Inserte los elementos en la lista, de tal manera que estos se inserten ordenados.
- Cree un menú de opciones que permita gestionar la lista de canciones de mi reproductor musical implantando las tareas básicas estudiadas.