TEMA 2

Estructuras de Datos Dinámicas

Contenido del Tema

2.1. Introducción a las estructuras de datos dinámicas.

2.3. Operaciones sobre listas enlazadas.

Introducción a las Estructuras de Datos Dinámicas

- Variables estáticas:
  - Se conoce su nombre.
  - Se conoce cuando empieza/acaba
  - .
  - espacio que ocupan

Se Crean en Tiempo de Compilación

Introducción a las Estructuras de Datos Dinámicas

Problema

¿Que sucede si a priori no conocemos la cantidad de espacio de almacenamiento que vamos a precisar?

⇔ Hacer una previsión??

Ejemplo:

```
Tipos
   TipoPoblacion = ARRAY[1..50] DE TipoPersona
Variables
    poblacion   : TipoPoblacion
```

Introducción a las Estructuras de Datos Dinámicas

- Variables anónimas:
  - No se conoce su nombre.
  - No se conoce el momento en que empieza/acaba su .
  - espacio que van a ocupar memoria es variable.

Se Crean en Tiempo de Ejecución

PUNTEROS

¿Que Es un Puntero?

Variable Estática

Un puntero es una variable que almacena una dirección de memoria

Declaración de Punteros

```
Tipos
   PtrATipo =PUNTERO A Tipo

Variables
   ptr: PtrATipo
```

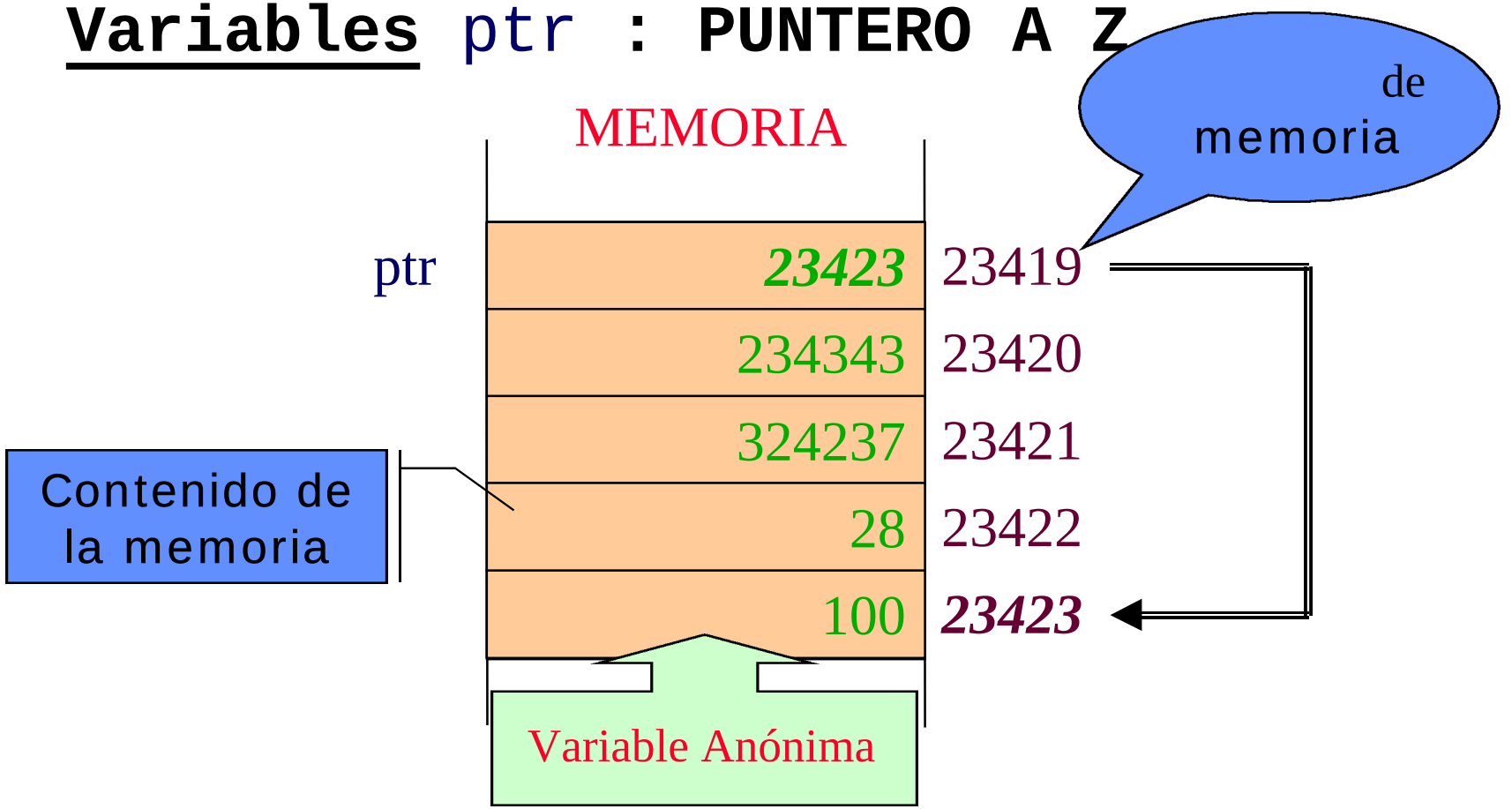
1ª FORMA

```
Variables
   ptr: PUNTERO A Tipo
```

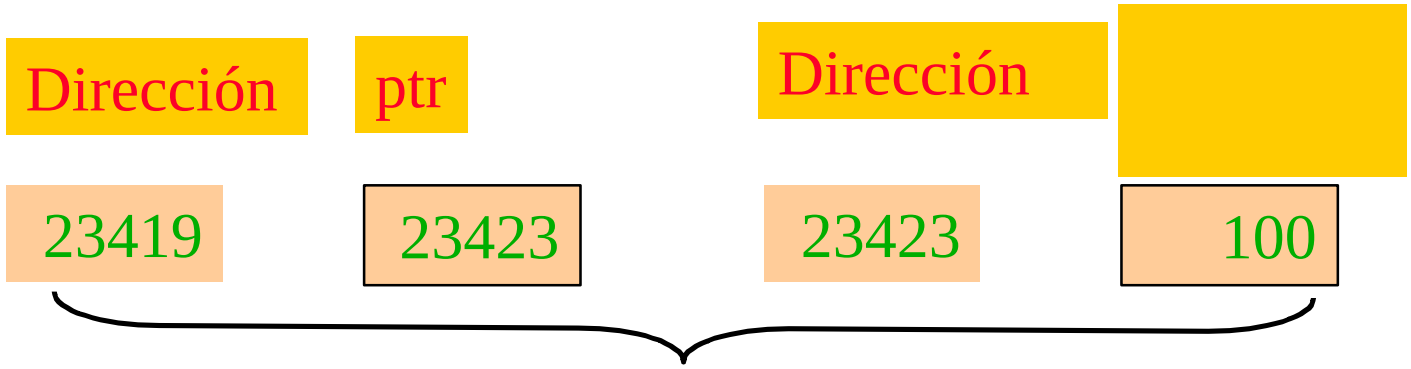
2ª FORMA

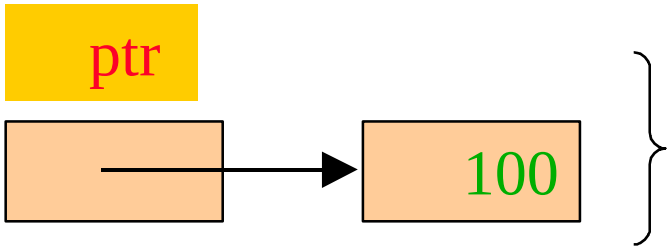
Variables Anónimas

Variables ptr : **PUNTERO A Z**



Representación Gráfica de las Variables Puntero

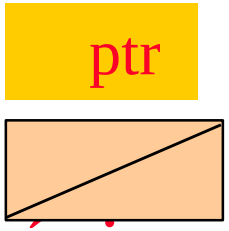




Punteros y Creación de Variables Dinámicas

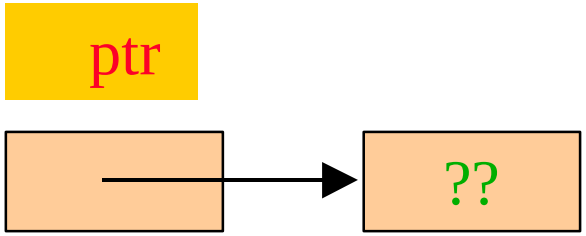
Inicialización de Punteros:

```
ptr :=  NIL
```



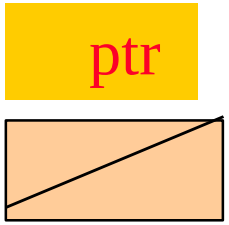
Creación de una Variable Anónima:

```
Asigna (ptr, Tamaño(Tipo))
```



Destrucción de una Variable Anónima:

```
Libera (ptr, Tamaño(Tipo))
```



Operaciones con Punteros

Dereferenciación: `ptr^`

Comparación: `ptr1 = ptr2`

Asignación: `ptr1 := ptr2`

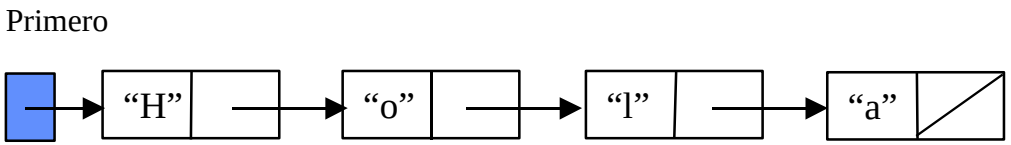
Ejemplo:

Asigna(ptr1, Tamaño(Z))	Escribir(ptr1^) ??
ptr1^:= 3	Escribir(ptr2^) ??
ptr2 := ptr1	
ptr2^:=1000	

Ejemplo: Cadena de Caracteres Dinámica

```
Tipos
     Tlista  = PUNTERO A INFO
     INFO =     REGISTRO
                   dato : $
                   sig : Tlista
                FIN
Variables     primero: Tlista
```

Gráficamente:



Ejemplo: Cadena de Caracteres Dinámica

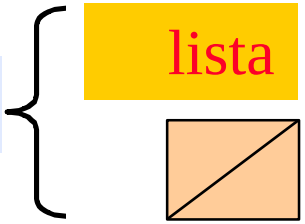
```
Algoritmo CrearLista():Tlista
Constantes
  RETORNOCARRO = CHR(13)
Variables
  car                 : $
  primero, ptr        : Tlista

Inicio
   Leer(car)
   SI car = RETORNOCARRO ENTONCES
       primero := NIL
   ENOTROCASO
     Asignar(primero,Tamaño(INFO))
     primero^.dato := car

          car)
    MIENTRAS car<>RETORNOCARRO HACER
     Asigna(ptr^.sig,Tamaño(INFO))
     ptr := ptr^.sig
     ptr^.dato := car
     Leer(car)
    FINMIENTRAS
    ptr^.sig := NIL
  FINSI
  DEVOLVER primero
Fin
```

Operaciones sobre Listas Enlazadas

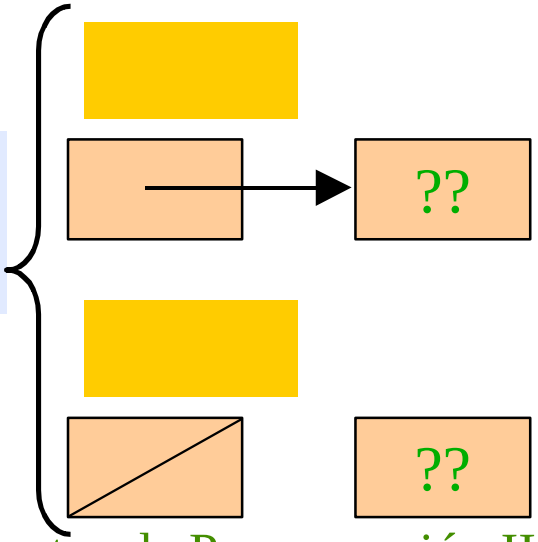
INICIALIZACION

Correcto

```
lista := NIL
```




```
Asignar(lista,Tamaño(INFO))
lista := NIL
```

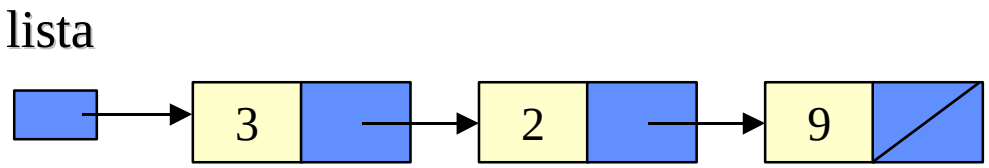


Visualizar una Lista

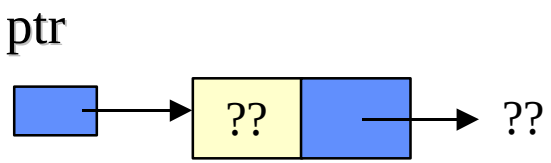
```
Algoritmo visualizaLista(lista: Tlista)
    Variables
        recorrer : Tlista
    Inicio
        recorrer := lista
        MIENTRAS recorrer <> NIL HACER
            Escribir(recorrer^.dato);
            recorrer := recorrer^.sig
        FINMIENTRAS
    Fin
```

Insertar un Nodo al Principio

Suponiendo:

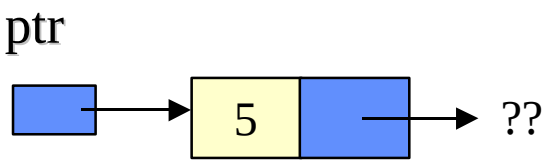



```
1.- Asigna (ptr, Tamaño(INFO))
```

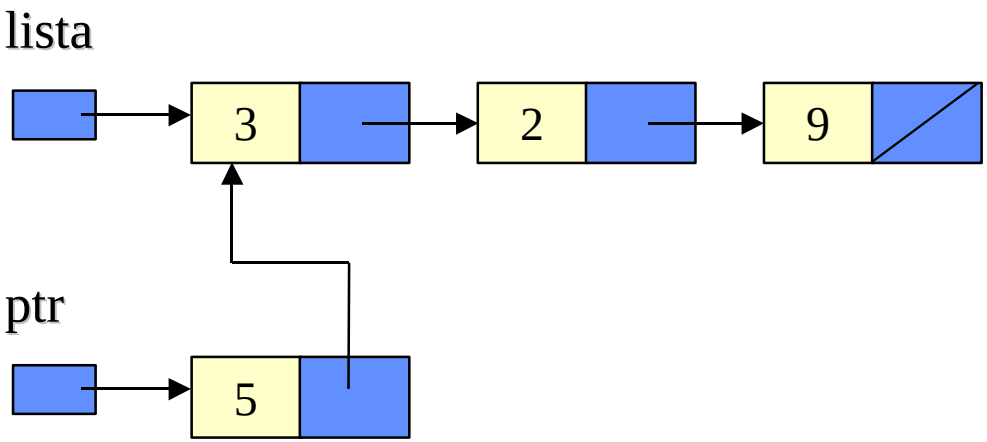


Insertar un Nodo al Principio

```
2.- ptr^.dato:= 5
```

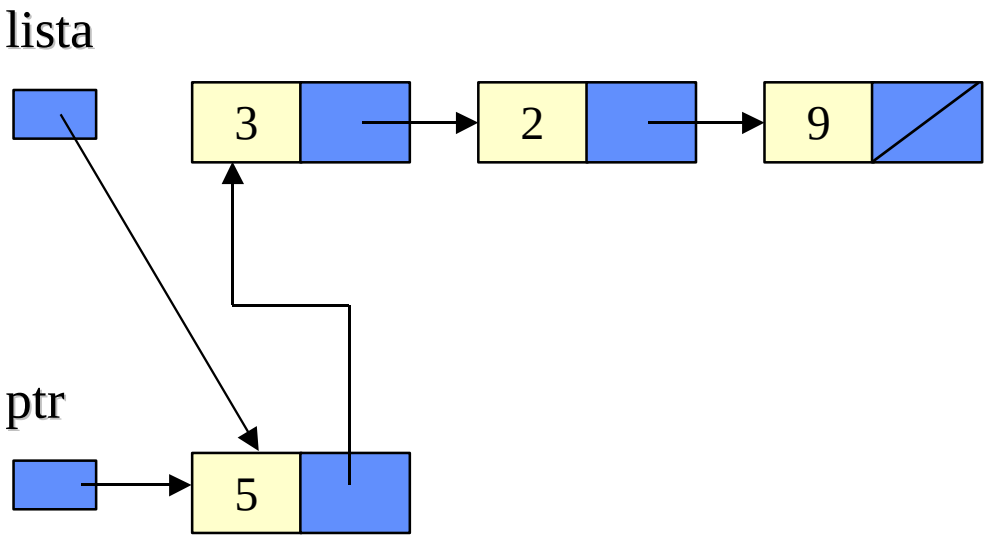



```
3.- ptr^.sig:= lista
```



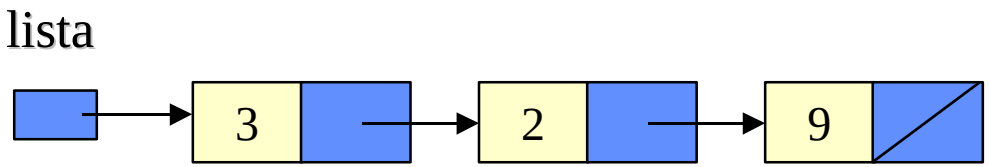
Insertar un Nodo al Principio

```
4.- lista:= ptr
```

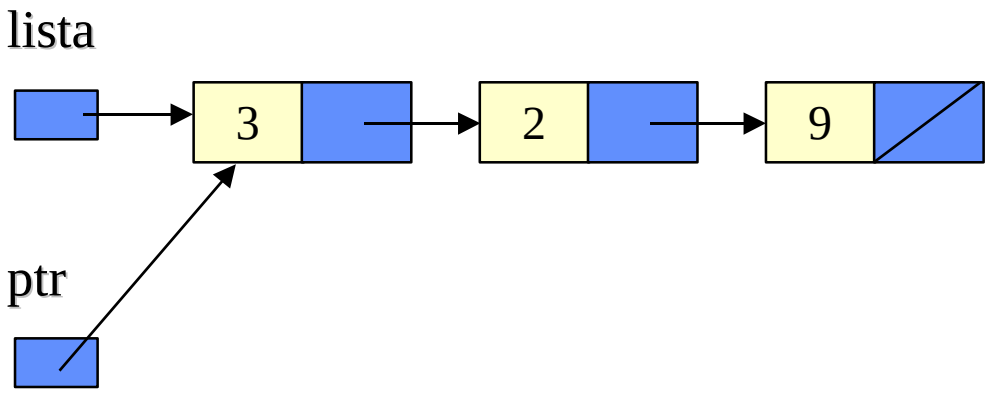


Eliminar el Primer Nodo

Suponiendo:

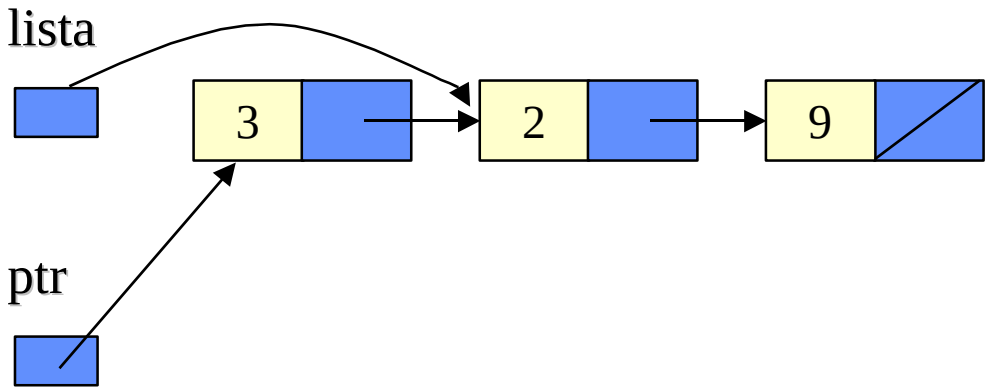



```
1.- ptr := lista
```

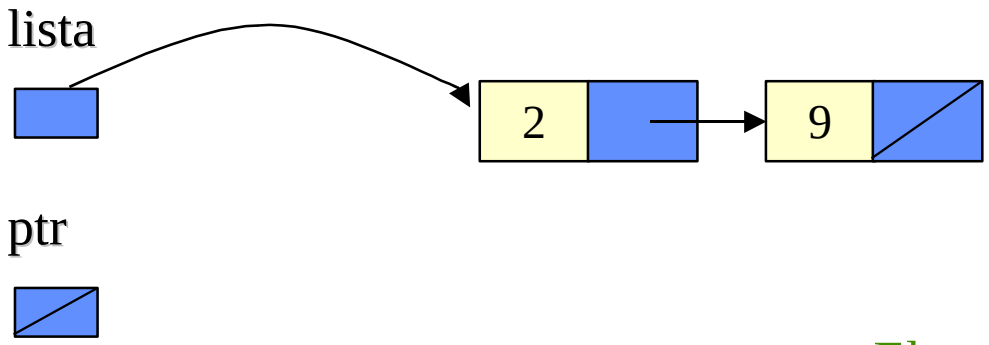


Eliminar el Primer Nodo

```
2.- lista := lista^.sig
```



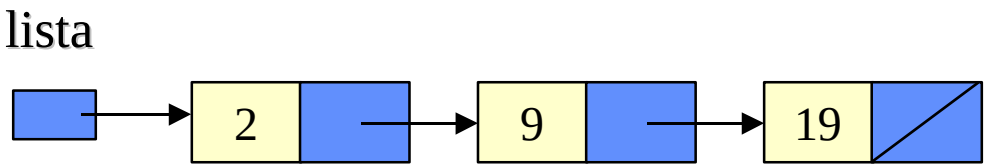

```
3.- Libera(ptr, Tamaño(INFO))
```



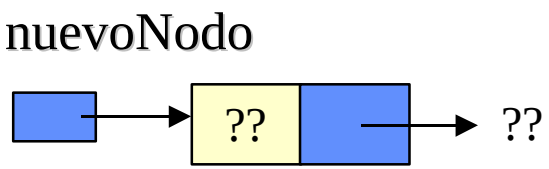
Insertar un Nodo en una Lista Enlazada Ordenada

Partimos de la lista:

Queremos insertar el número: 15

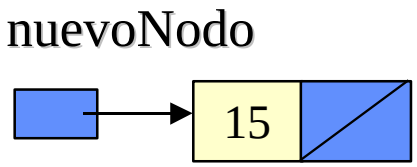



```
1.- Asigna(nuevoNodo, Tamaño(INFO))
```



Insertar un Nodo en una Lista Enlazada Ordenada

```
2.- nuevoNodo^.dato := 15
    nuevoNodo^.sig  := NIL
```



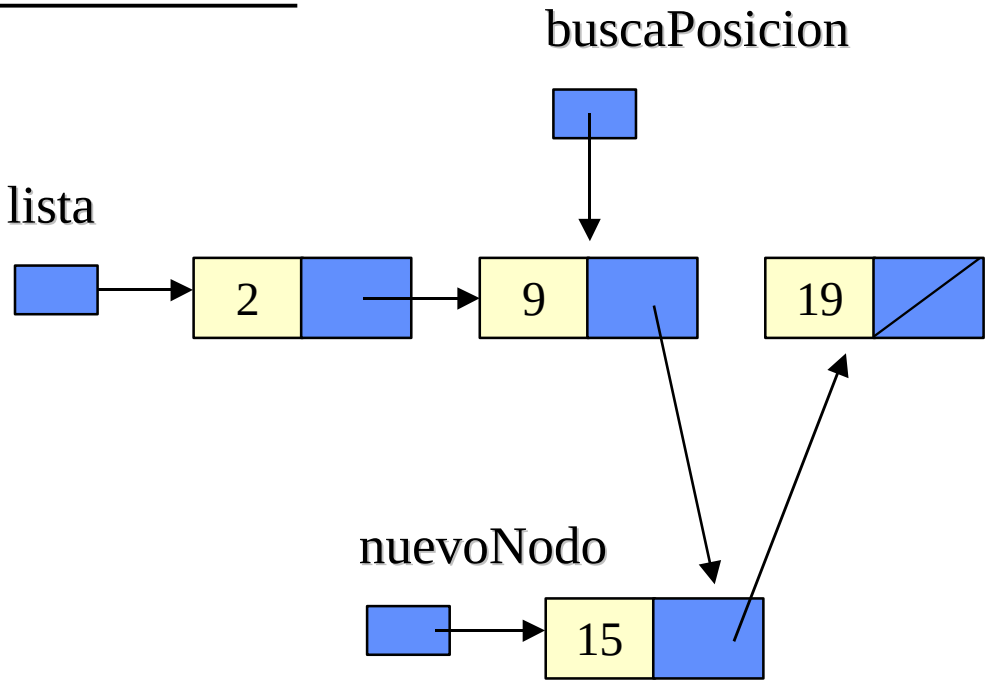

```
3.- Algoritmo que inserta el nodo
    en la posición correcta.
```

Insertar un Nodo en una Lista Enlazada Ordenada

```
Algoritmo InsertaNuevoNodo( VAR lista: Tlista; dato: Z)
Variables
 nuevoNodo    : Tlista
 buscaPosicion: Tlista
Inicio
 Asigna(nuevoNodo, Tamaño(INFO))
 nuevoNodo^.dato:= dato
 nuevoNodo^.sig := NIL
 SI lista = NIL ENTONCES
    (* Si la lista está vacía *)

 ENOTROCASO
   SI (nuevoNodo^.dato <=
         lista^.dato) ENTONCES
       nuevoNodo^.sig:= lista
       lista := nuevoNodo
 ENOTROCASO
    buscaPosicion := lista
  MIENTRAS ((buscaPosicion^.sig <> NIL)
       AND (nuevoNodo^.dato >=
       buscaPosicion^.sig^.dato)) HACER
    buscaPosicion := buscaPosicion^.sig
  FINMIENTRAS
  nuevoNodo^.sig := buscaPosicion^.sig
  buscaPosicion^.sig := nuevoNodo;
 FINSI
FINSI
Fin
```

Insertar un Nodo en una Lista Enlazada Ordenada

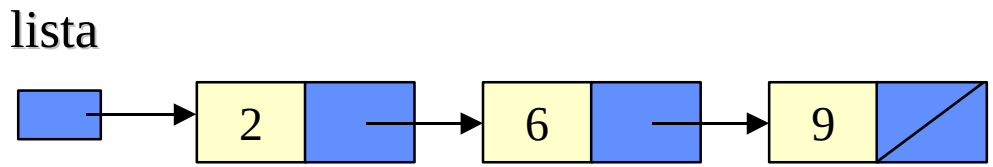
Gráficamente:



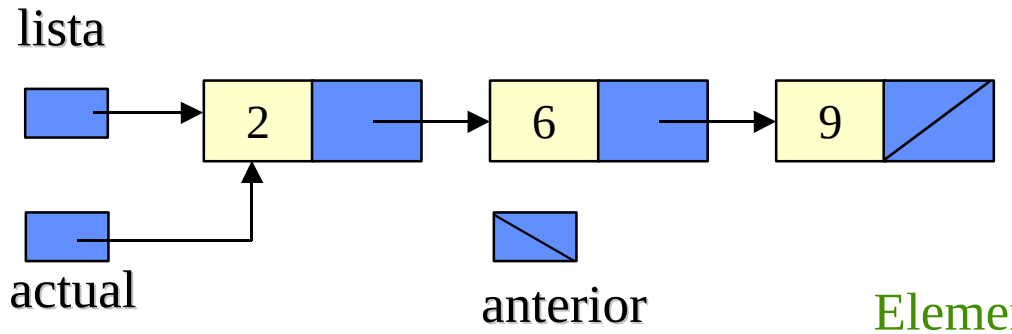
Eliminar un Nodo de una Lista Enlazada

Suponiendo:

Queremos borrar el número: 6
Supondremos que está en la lista.



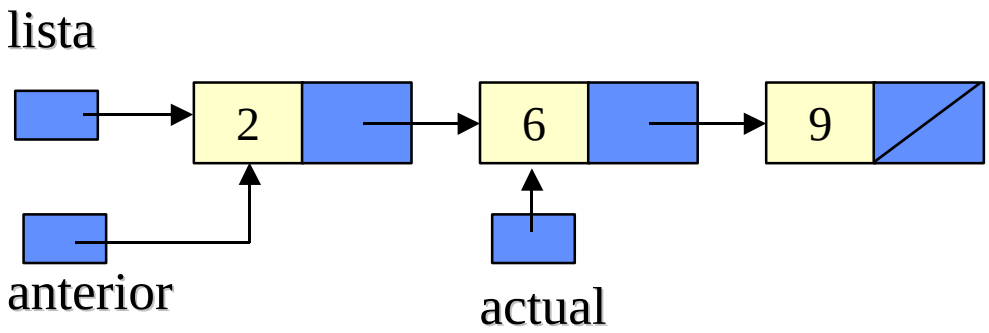

```
1.- actual   := lista
    anterior := NIL
```



Eliminar un Nodo de una Lista Enlazada

2.- Búsqueda del nodo a borrar.

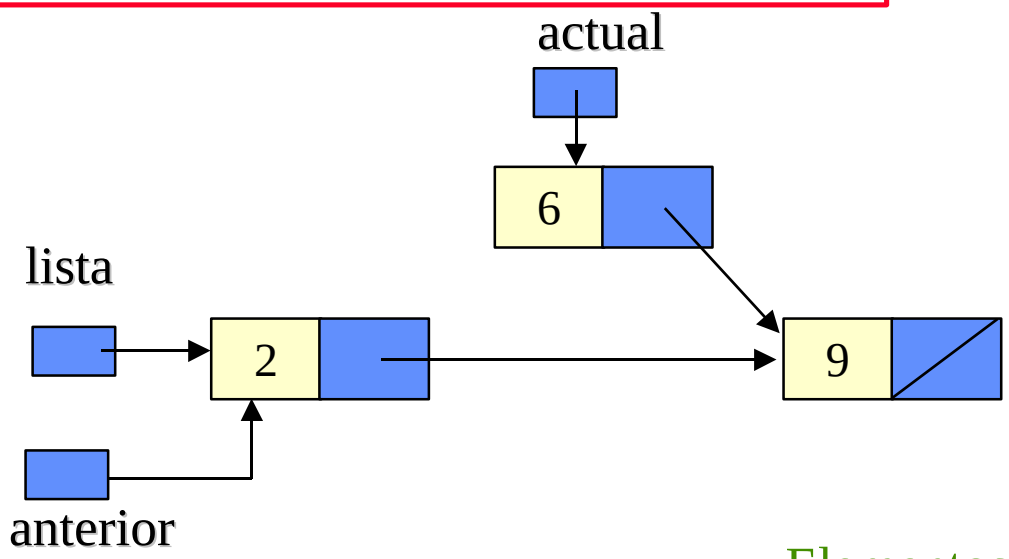
```
MIENTRAS actual^.dato <> dato HACER
    anterior := actual
    actual:= actual^.sig
FINMIENTRAS
```



Eliminar un Nodo de una Lista Enlazada

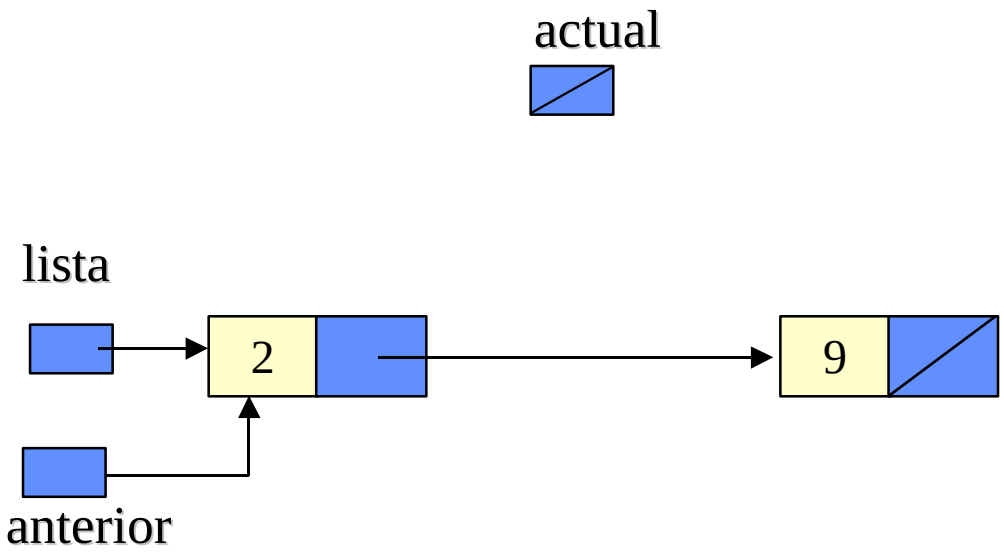
3.- Actualizar los punteros

```
SI anterior = NIL ENTONCES
   lista := lista^.sig
ENOTROCASO
   anterior^.sig := actual^.sig
FINSI
```



Eliminar un Nodo de una Lista Enlazada

4.- Libera (actual, Tamaño(INFO))



Eliminar Todos los Nodos de una Lista Enlazada

Algoritmo borrarLista(**VAR** lista: Tlista)

Variables

nodoABorrar: Tlista

Incio

MIENTRAS lista <> **NIL HACER**

nodoABorrar := lista;

lista := lista^.sig

Libera (

FINMIENTRAS

Fin

BIBLIOGRAFIA

- **Pascal**. Dale/Orshalick. Ed. McGraw Hill 1986
- **Programación I**. José A. Cerrada y Manuel Collado. U.N.E.D. 1993.
- **Fundamentos de Programación**. Joyanes Aguilar. McGraw Hill. 1988.
- **Introduction to programming with** . Saim Ural/Suzan Ural. Wiley. 1987