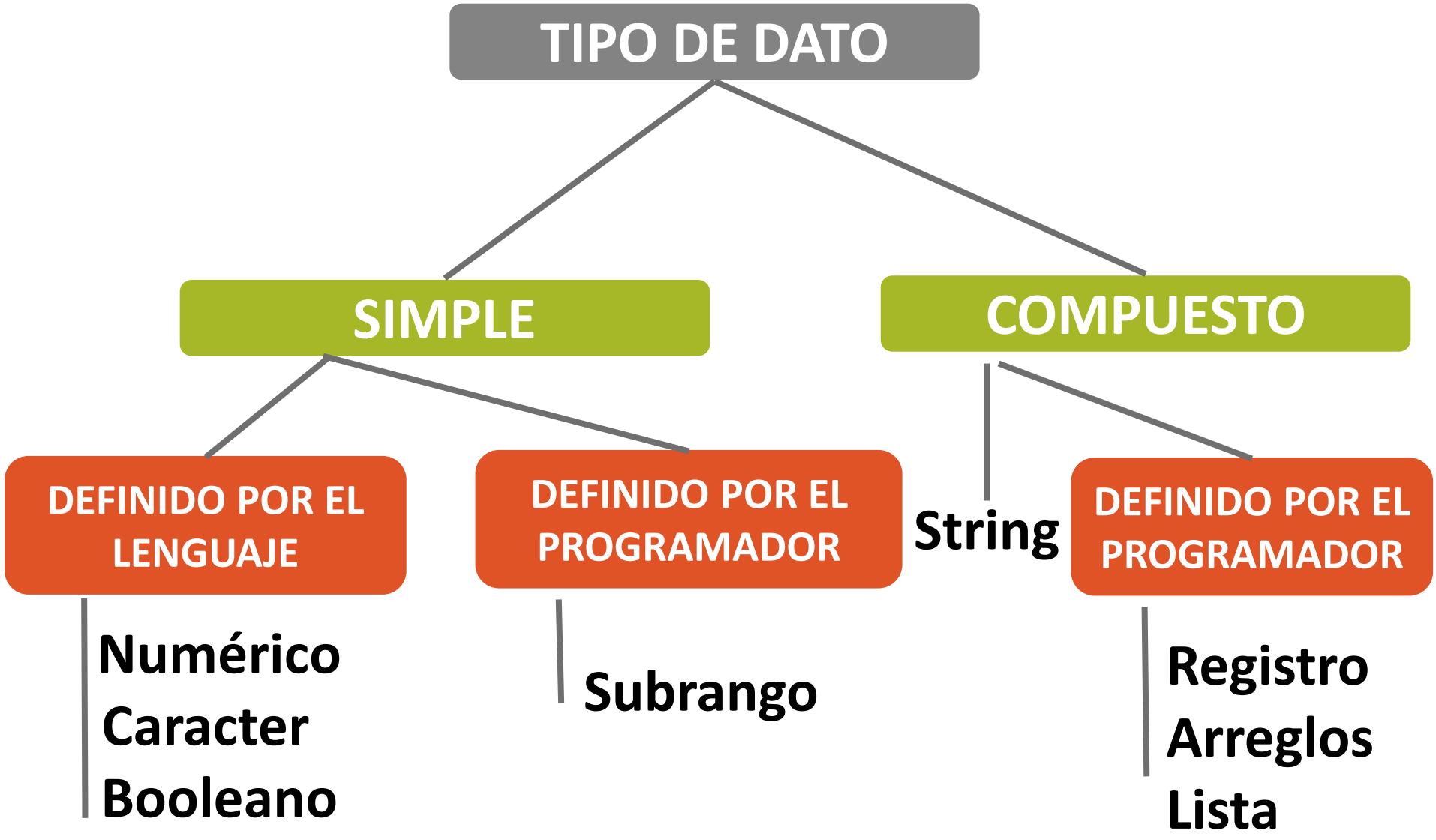
Conceptos de Algoritmos Datos y Programas

CADP – Temas de la clase de hoy

- Tipo de dato lista
- Características de las listas
- Operaciones
- Ejercicios
- Comparación con la estructura arreglos

CADP – Tipos de Datos



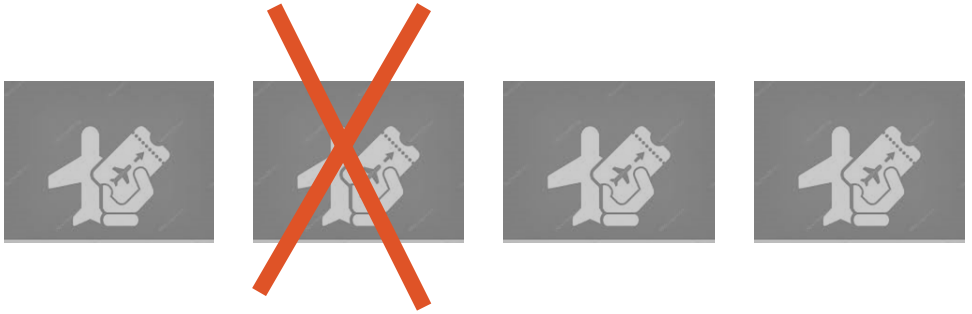
CADP – Tipos de Datos – Lista - Motivación

Supongamos que estamos trabajando en la página de una aerolínea para la reserva de pasajes.

Con las estructuras que conocemos, donde se almacenan las reservas?

PROBLEMAS?

Hay reservas que se confirman y reservas que se cancelan?



CADP – Tipos de Datos – Lista - Motivación

El concepto de lista es bastante intuitivo. Existe y se ve en un montón de situaciones de la vida cotidiana

- Lista de alumnos
- Lista de autos
- Lista de productos de un supermercado

CADP – Tipos de Datos – Lista - Definición

Estructura de Lista

Colección de elementos homogéneos, con una relación lineal que los vincula, es decir que cada elemento tiene un único predecesor (excepto el primero), y un único sucesor (excepto el último).

Los elementos que la componen pueden no ocupar posiciones contiguas de memoria. Es decir pueden aparecer dispersos en la memoria, pero mantienen un orden lógico interno.

num: 536

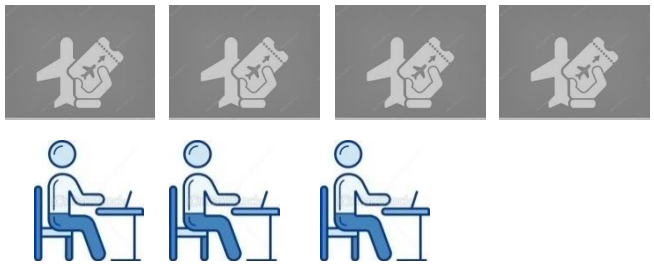
pun: FAFB

145	FFFB	AAFA
58	AAFA	FAFB
32	nil	FFFB

CADP – Estructura de Datos – Lista - Características

Homogénea

- Los elementos son todos del mismo tipo



Dinámica

- La cantidad de elementos puede variar durante la ejecución

Lineal

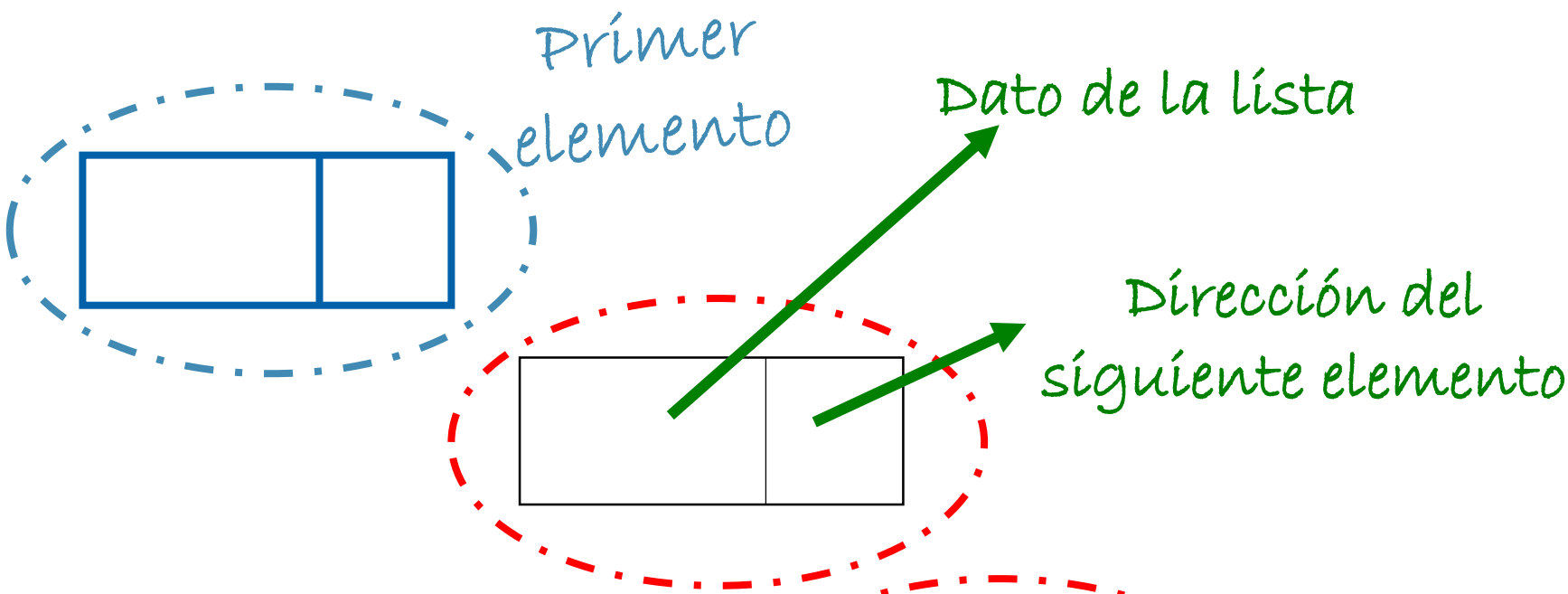
- Cada elemento tiene un único antecesor y sucesor

CADP – Estructura de Datos – Lista - Características

- Está compuesta por NODOS.
- Los nodos se conectan por medio de enlaces o punteros.
- Cuando se necesita espacio adicional, nuevos nodos pueden ser alocados y agregados a la estructura (NEW).
- Cuando existen nodos que ya no se necesitan, pueden ser borrados, liberando memoria (DISPOSE).

cómo se ve gráficamente?

CADP – Estructura de Datos – Lista - Carcterísticas



Cómo se ve una lista de alumnos?

CADP – Estructura de Datos – Lista - Carcterísticas

María 3568/7	BCDA

Dirección Inicial

Juan 1548/0	ACFB

Dirección BCDA

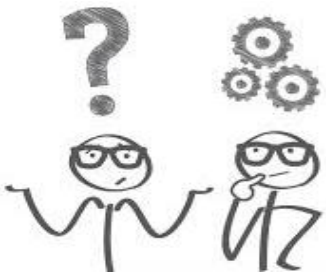
Morena 2248/5	CCFB

Dirección ACFB

Lucas 4242/8	nil

Dirección CCFB

Por qué el último elemento tiene nil en el siguiente?



Cómo se declara?

CADP – Estructura de Datos – Lista - Declaración

```
Program uno;
Type nombreTipo= ^nombreElemento;

     nombreElemento= Record
                       elementos : tipoElemento;
                       punteroSig: nombreTipo;
                     end;

Var
  Pri: nombreTipo; {Memoria estática reservada}
```

va primero

Cualquiera de los tipos vistos

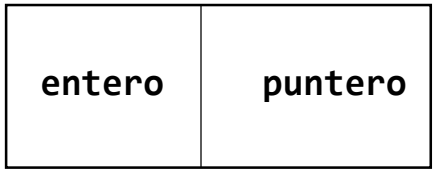
Estructura recursiva

Lista de enteros?

Lista de alumnos?

CADP – Estructura de Datos – Lista - Declaración

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
                    elementos : integer;
                    sig: listaE;
                   end;

Var
  PriEnteros: listaE; {Memoria estática reservada}
```



CADP – Estructura de Datos – Lista - Declaración

```
Program uno;
Type
     alumno = record
                nombre:string;
                numeroA:string;
              end;

     listaA= ^datosAlumnos;

     datosAlumnos= Record
                    elementos:alumno;
                    sig: listaA;
                   end;
Var
  PriAlumnos: listaA; {Memoria estática reservada}
```

alumno	puntero

CADP – Estructura de Datos – Lista – Operaciones

Crear una lista vacía

Agregar elementos (al inicio, al final)

Insertar elementos en una lista

Recorrer una lista

Recorrer hasta el i-esimo elemento

Borrar un elemento en una lista

Combinar dos listas ordenadas (Merge)

Trabajaremos con una lista de enteros

CADP – Estructura de Datos – Lista – Operaciones

Crear una lista vacía. **Significa indicar que la lista no tiene dirección de comienzo asignado.**

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
  PriE: listaE;
```

```
Begin
  PriE:= nil;
End.
```

Por qué no se hace new(priE)?

Modularizado?

CADP – Estructura de Datos – Lista – Operaciones

Crear una lista vacía. **Significa indicar que la lista no tiene dirección de comienzo asignado.**

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Procedure crearLista (var p: listaE);
 begin
   p:= nil;
 end;

Var
   priE:listaE;

Begin
 crearLista(priE);
End.
```

CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al inicio. Significa indicar que por cada elemento a agregar se debe modificar el puntero inicial de la lista.

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;
```

```
Begin
  PriE:= nil;
  read (valor);
  agregarAdelante(priE,valor);
End.
```

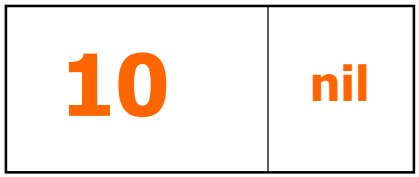
Qué debo hacer?

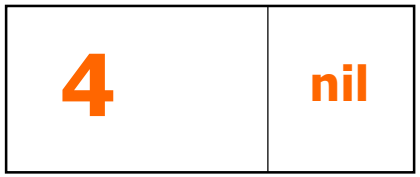
CADP – Estructura de Datos – Lista – Operaciones

Supongamos que se lee *10 4 25*

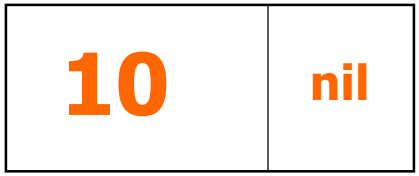
PriE:= nil;

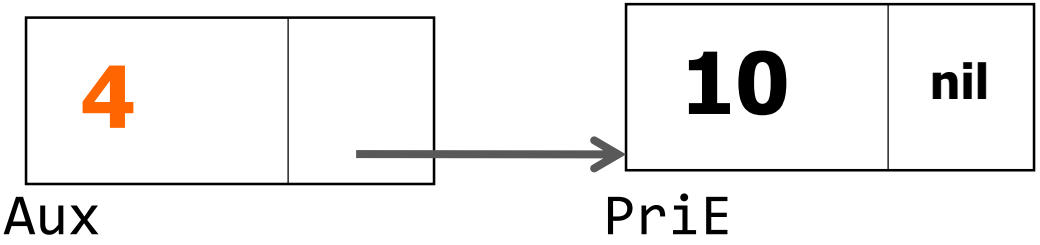
Como priE es nil, entonces es el primer elemento





Como priE NO es nil, entonces tengo que actualizar enganches y PriE

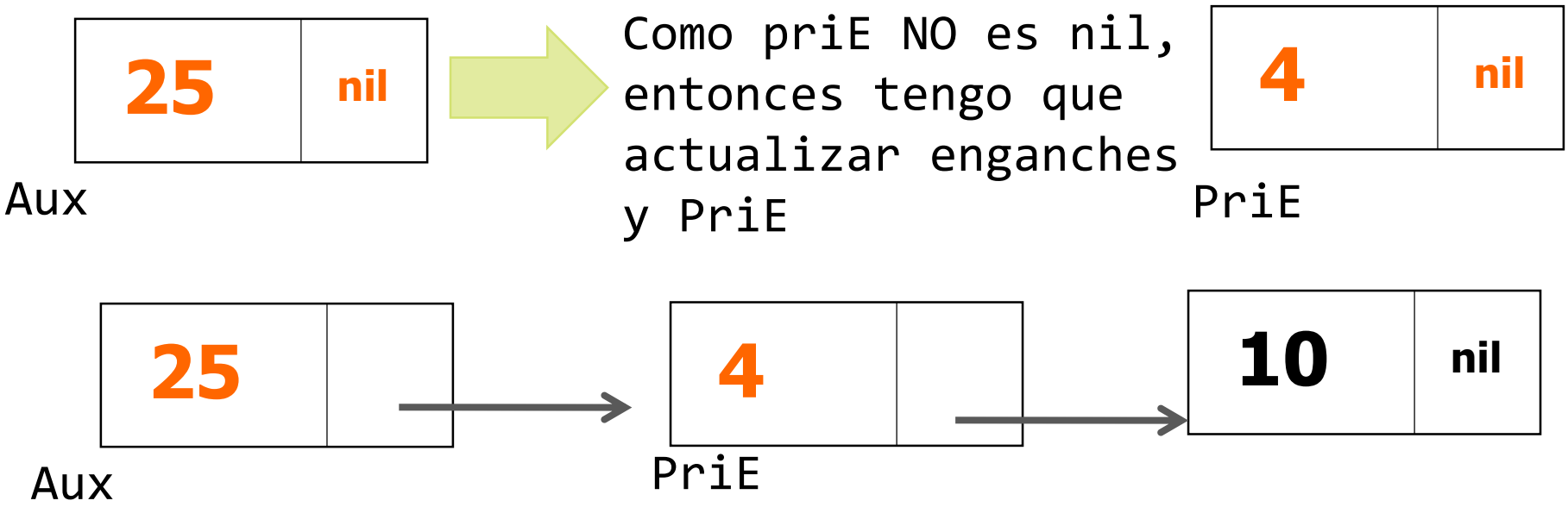




CADP – Estructura de Datos – Lista – Operaciones

Supongamos que se lee *10 4 25*

PriE:= nil;



CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al inicio. Significa indicar que por cada elemento a agregar se debe modificar el puntero inicial de la lista.

Cómo lo escribo?

```
Reservo espacio en memoria.

si (es el primer elemento a agregar)
  asigno el puntero inicial.

sino
  actualizo el puntero al siguiente del nuevo
  elemento.
  actualizo el puntero inicial de la lista
```

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarAdelante (var p:listaE
                           valor:integer);
Var
  aux:listaE;

Begin
  new (aux); aux^.elem:= valor; aux^.sig:=nil;

  if (p = nil) then p:= aux
  else begin
    aux^.sig:= p;
    p:=aux;
  end;
End;
```

Creo espacio para el nuevo elemento

Evalúo el caso y reasigno los punteros

CADP – Estructura de Datos – Lista – Operaciones

Recorrer una lista. **Significa poder avanzar por cada uno de los elementos de la lista.**

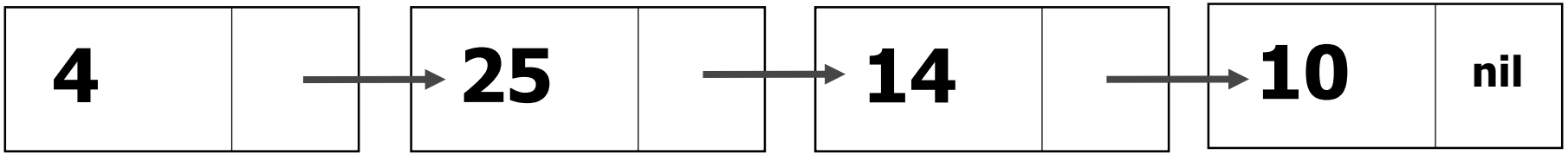
```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;
```

```
Begin
  PriE:= nil;
  read (valor);
  agregarAdelante(priE, valor);
  imprimirLista(priE);
End.
```

Qué debo hacer?

CADP – Estructura de Datos – Lista – Operaciones

Recorrer una lista. **Significa poder avanzar por cada uno de los elementos de la lista.**



Pri

aux

4 25 14 10

Qué debo hacer?

CADP – Estructura de Datos – Lista – Operaciones

Recorrer una lista. Significa poder avanzar por cada uno de los elementos de la lista.

Cómo lo escribo?

```
Inicializo una variable auxiliar

mientras (no sea el final de la lista)

    proceso el elemento (por ejemplo imprimo)
    avanzo el puntero auxiliar
```

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure imprimirLista (p:listaE);
Var
   aux:listaE;

Begin
   aux:= p;
   while  (aux <> nil) do
      begin
         write (aux^.elem);
         aux:= aux^.sig;
      end;
End;
```

Inicializo el recorrido

Imprimo el elemento

Avanzo el puntero

Está mal escribir aux^.sig <> nil?

Necesito la variable aux?

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure imprimirLista (p:listaE);

Begin
  while  (p <> nil) do
    begin
      write (p^.elem);
      p:= p^.sig;
    end;
End;
```

CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al final. **Significa que debo recorrer la lista y llegar hasta el final y reacomodar los punteros.**

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;

Begin
   PriE:= nil;
   read (valor);
   agregarAlFinal(priE, valor);
End.
```

Qué debo hacer?

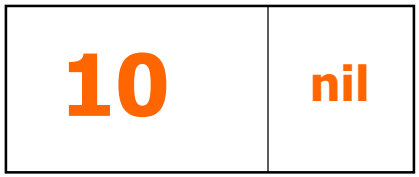
CADP – Estructura de Datos – Lista – Operaciones

Supongamos que se lee *10 4 25*

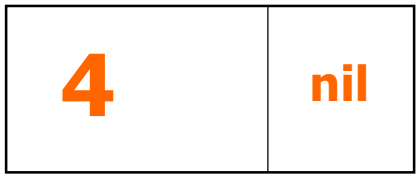
PriE:= nil;

Aux

Como priE es nil, entonces es el primer elemento

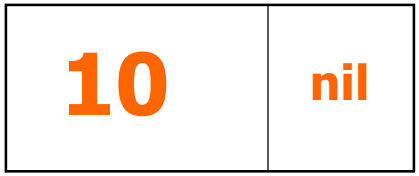


PriE



Aux

Como priE NO es nil, entonces tengo que llegar al final de la lista y actualizar los punteros



PriE

PriE

Pos

Aux

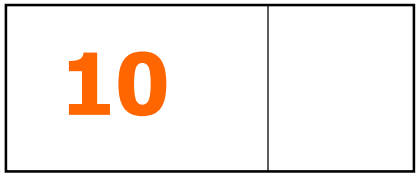
CADP – Estructura de Datos – Lista – Operaciones

Supongamos que se lee *10 4 25*

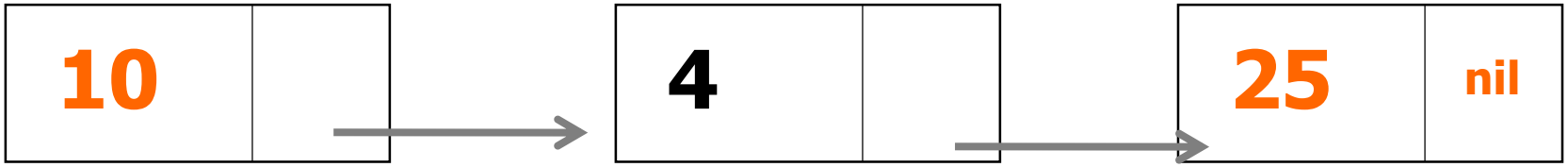
PriE:= nil;

Aux

Como priE NO es nil, entonces tengo que llegar al final de la lista y actualizar los punteros



PriE



PriE

Pos

Aux

CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al final. **Significa que debo recorrer la lista y llegar hasta el final y reacomodar los punteros.**

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;

Begin
  PriE:= nil;
  read (valor);
  agregarAlFinal(priE, valor);
End.
```

CADP – Estructura de Datos – Lista – Operaciones

Agregar al final.

Cómo lo escribo?

Genero espacio de memoria y cargo los datos *aux*

Inicializo un puntero auxiliar *pos*

mientras (no se llegue al último elemento)

**   avanzo el puntero pos**

Reacomodo los punteros

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarAlFinal (var p:listaE;valor:integer);
Var
  aux,pos:listaE;

Begin
  new (aux); aux^.elem:=valor;   aux^.sig:=nil;

  if (p = nil) then
    p:= aux
  else begin
    pos:= p;
    while (pos^.sig <> nil) do
      pos:= pos^.sig;
    end;
    pos^.sig:= aux;
  end;
End;
```

Otra forma?

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarAlFinal (var p:listaE valor:integer);
Var
  aux,pos,ant:listaE;

Begin
  new (aux); aux^.elem:= valor; aux^.sig:=nil;
  if (p = nil) then
    p:= aux
  else begin
    pos:= p; ant:=p;
    while (pos <> nil) do begin
      ant:=pos;
      pos:= pos^.sig;
    end;
    ant^.sig:= aux;
  end;
End;
```

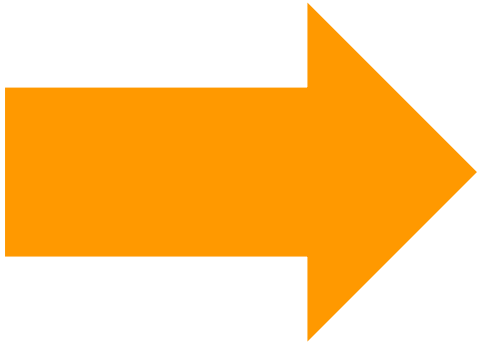
Opción 1 para agregar pero llevando un anterior

Se puede mejorar?

CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al final. **Significa que debo recorrer la lista y llegar hasta el final y reacomodar los punteros.**

Opción 1



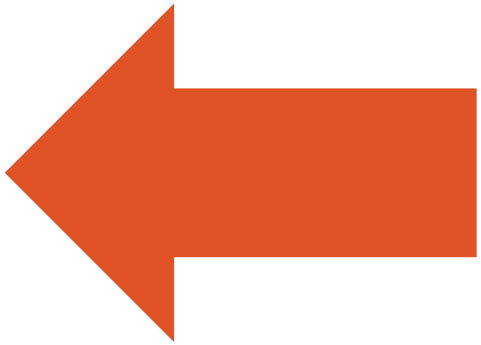
Genero espacio para el nuevo elemento

Recorro la lista hasta llegar al último elemento.

Reasigno los punteros.

Recién implementada

Opción 2



Genero espacio para el nuevo elemento

Utilizo un puntero que mantiene la dirección del último elemento.

Reasigno los punteros.

Reasigno la dirección del último elemento

CADP – Estructura de Datos – Lista – Operaciones

Agregar un elemento al final (opción 2).

Significa que debo llevar un puntero que marque el último elemento de la lista la lista y reacomodar los punteros.

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE,ult: listaE;
 valor:integer;

Begin
  PriE:= nil;
  read (valor);
  agregarAlFinal2(priE,ult,valor);
End.
```

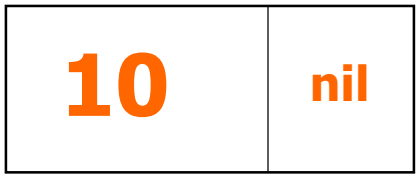
CADP – Estructura de Datos – Lista – Operaciones

Supongamos que se lee *10 4 25*

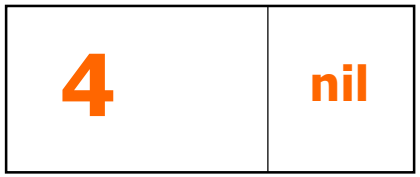
PriE:= nil;

Aux

Como priE es nil, entonces es el primer elemento

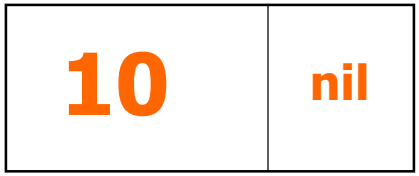


PriE
UltE



Aux

Como priE NO es nil, entonces tengo que actualizar el siguiente al último y luego actualizar el último



PriE
UltE

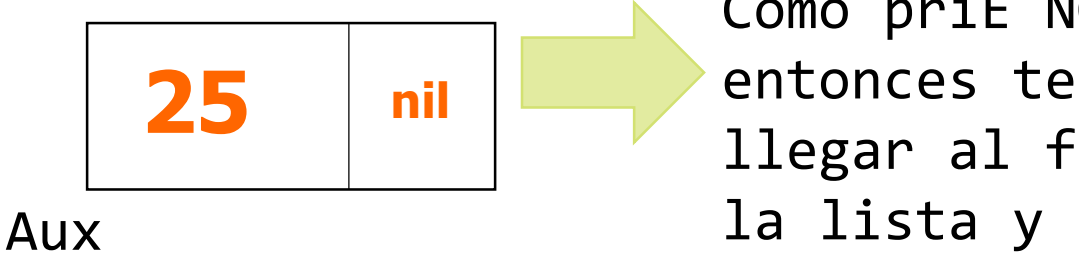
PriE
ultE

Aux

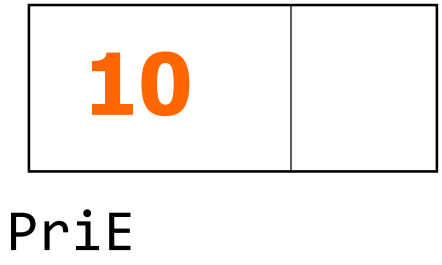
CADP – Estructura de Datos – Lista – Operaciones

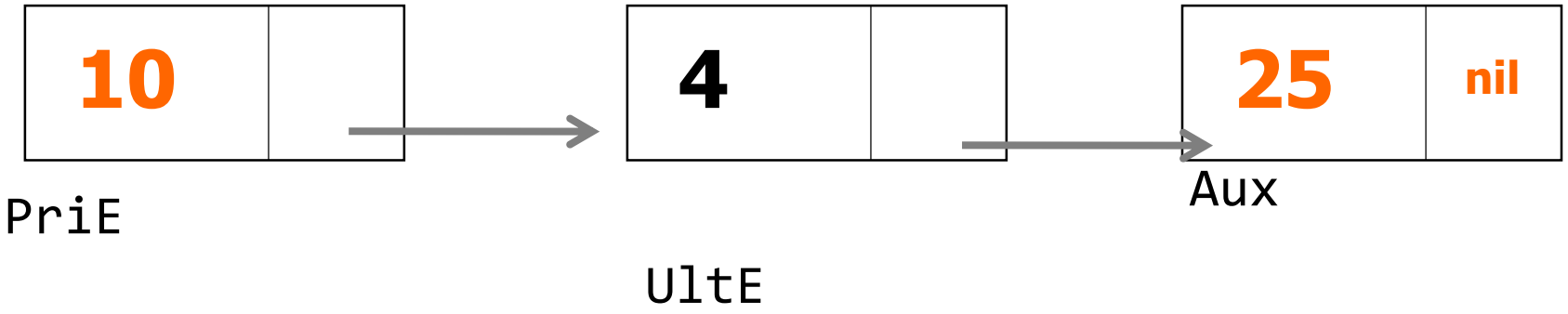
Supongamos que se lee *10 4 25*

PriE:= nil;



Como priE NO es nil, entonces tengo que llegar al final de la lista y actualizar los punteros





CADP – Estructura de Datos – Lista – Operaciones

Agregar al final (opción 2).

Cómo lo escribo?

```
Genero espacio de memoria y cargo los datos aux

si (es el primer elemento)
 asigno el primer y último puntero de la lista

sino
 actualizo el puntero siguiente del último elemento
 actualizo el último elemento
```

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarAlFinal2 (var p,ult:listaE;
                           valor:integer);
Var
  aux:listaE;
Begin
  new (aux); aux^.elem:= valor; aux^.sig:=nil;

  if (p = nil) then begin
    p:= aux; ult:=aux;
  end
  else begin
        ult^.sig:= aux;
        ult:= aux;
       end;
End;
```

Por qué ult tiene que ser parámetro?

CADP – Estructura de Datos – Lista – Operaciones

Insertar ordenado. Significa indicar que por cada elemento a agregar se debe agregar en el lugar correspondiente para que la lista siga manteniendo un orden

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;
```

```
Begin
  PriE:= nil;
  read (valor);
  insertar(priE,valor);
End.
```

Qué debo hacer?

CADP – Estructura de Datos – Lista – Operaciones

Insertar ordenado.

Qué hago en cada caso?

Existen 4 casos:

1. La lista está vacía.
2. El elemento a insertar va al comienzo de la lista.
3. El elemento a insertar va al medio de la lista.
4. El elemento a insertar va al final de la lista.

PriE

CADP – Estructura de Datos – Lista – Operaciones

Insertar ordenado.

Existen 4 casos: CASO 1 , lista vacía PriE=nil

4	dir

aux

PriE

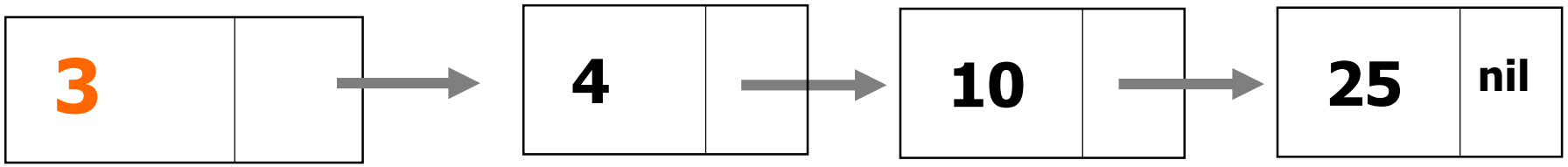
1.Genero espacio para el nuevo elemento

2.Asigno como dirección inicial de la lista este nuevo espacio

CADP – Estructura de Datos – Lista – Operaciones

Insertar ordenado.

Existen 4 casos: CASO 2 , el elemento va al principio



aux

priE

actual

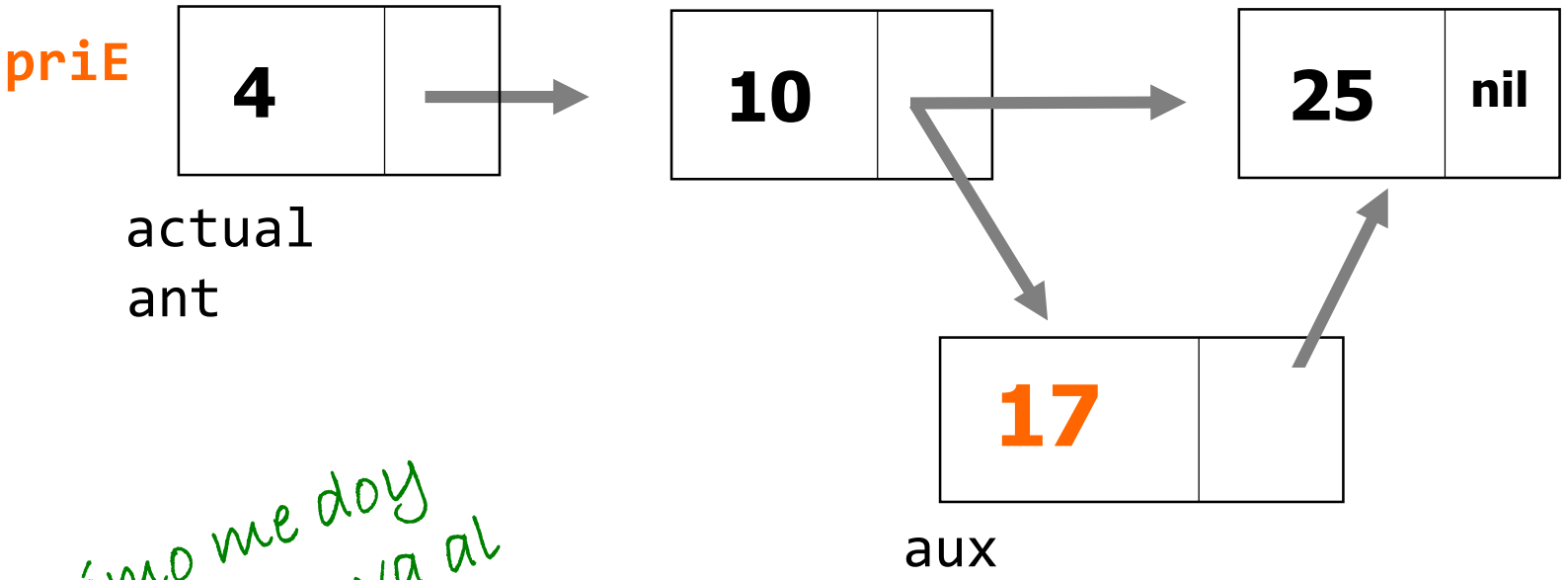
ant

Cómo me doy cuenta que tiene que insertarse primero?

1. Genero espacio para el nuevo elemento
2. Preparo los punteros para el recorrido.
3. Reorganizo punteros.
4. Actualizo la dirección del primer puntero.

Insertar ordenado.

Existen 4 casos: CASO 3 , el elemento va en el medio



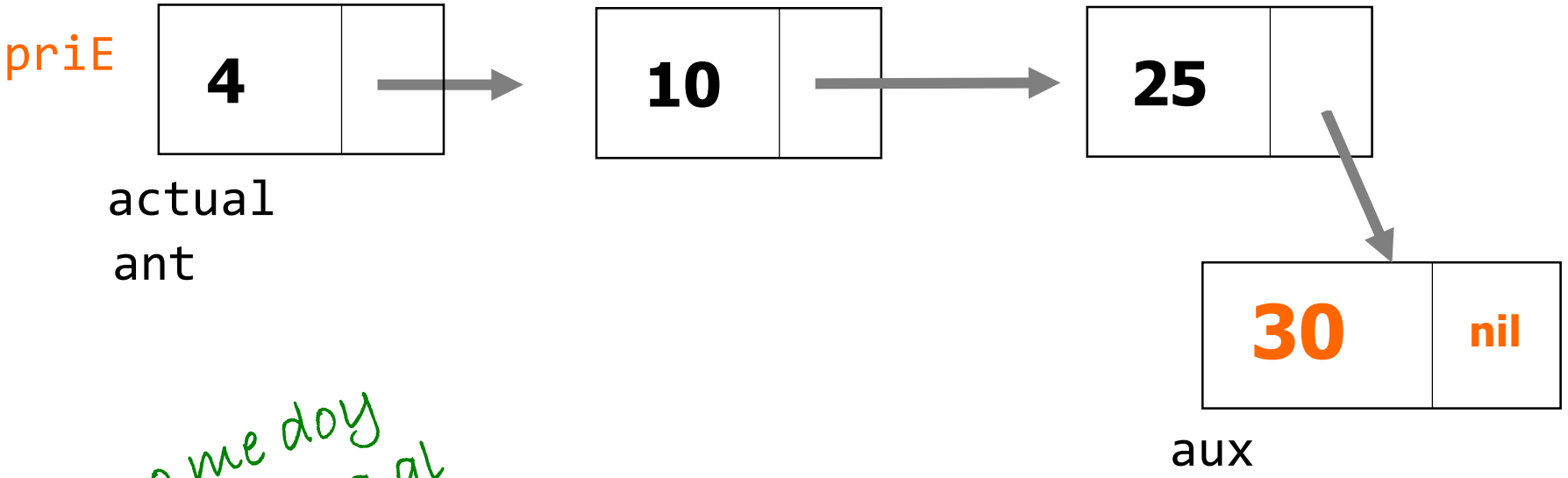
Cómo me doy cuenta que va al medio?

1. Genero espacio para el nuevo elemento
2. Preparo los punteros para el recorrido.
3. Reorganizo punteros.
4. Actualizo la dirección del primer puntero.(NO)

CADP – Estructura de Datos – Lista – Operaciones

Insertar ordenado.

Existen 4 casos: CASO 4 , el elemento va al fina



cómo me doy cuenta que va al final?

1. Genero espacio para el nuevo elemento
2. Preparo los punteros para el recorrido.
3. Reorganizo punteros.
4. Actualizo la dirección del primer puntero.(NO)

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarOrdenado(var p:listaE valor:integer);
Var
  aux,act,ant:listaE;

Begin
  new (aux); aux^.elem:= valor; aux^.sig:=nil;
  if (p = nil) then
    p:= aux
  else begin
    act:= p; ant:=p;

    while (act <> nil) and (act^.elem < aux^.elem) do
     begin
      ant:=act;
      act:= act^.sig;
     end;
  end;
```

Caso 1

Busco la posición

Verifico que caso es (2,3,4)

CADP – Estructura de Datos – Lista – Operaciones

```
if (act = p) then
 begin
   aux^.sig:= p;
   p:= aux;
 end
else if (act <> nil) then
 begin
  ant^.sig:=aux;
  aux^.sig:= act;
 end
else
  begin
   ant^.sig:= aux;
   aux^.sig:= nil;
  end;
End;
```

El elemento va al principio

caso 2

El elemento va al medio

caso 3

El elemento va al final

caso 4

Una solución mejor?

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure agregarOrdenado(var p:listaE valor:integer);
Var
  aux,act,ant:listaE;
Begin
  new (aux); aux^.elem:= valor; aux^.sig:=nil;

  if (p = nil) then
    p:= aux

  else begin
    act:= p; ant:=p;
    while (act <> nil) and (act^.elem<aux^.elem) do begin
      ant:=act;   act:= act^.sig;
    end;
  end;

if (act= p) then
 begin
   aux^.sig:= p;     p:= aux;
 end

else
 begin
  ant^.sig:=aux;    aux^.sig:= act;
 end
End;
```

CADP – Estructura de Datos – Lista – Operaciones

Eliminar un elemento. Significa recorrer la lista y cada vez que el elemento de la lista coincida con el elemento a eliminar realizar un dispose.

```
Program uno;

Type listaE= ^datosEnteros;

     datosEnteros= Record
         elem:integer;
         sig:listaE;
     end;

Var
 PriE: listaE;
 valor:integer;
```

```
Begin
  PriE:= nil;
  read (valor);
  eliminar(priE, valor);
End.
```

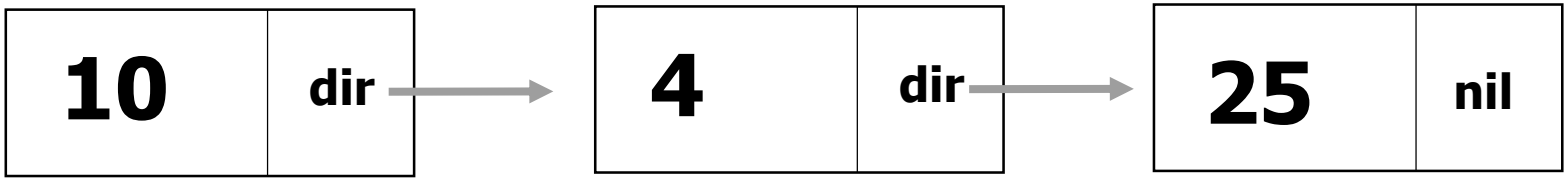
Qué debo hacer?

Eliminar un elemento.

Qué hago en cada caso?

Existen 4 casos:

1. La lista está vacía.
2. El elemento a eliminar está al comienzo.
3. El elemento a eliminar no está al comienzo.
4. El elemento a eliminar no está.



PriE

Elemento = 4

Elemento = 10

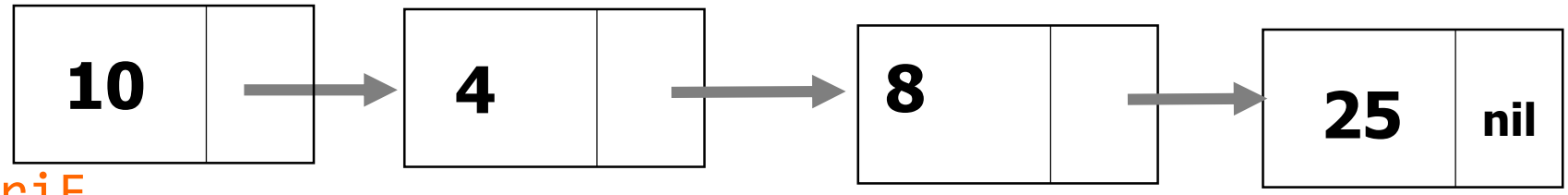
Elemento = 2

CADP – Estructura de Datos – Lista – Operaciones

Eliminar un elemento.

Existen 4 casos: CASO 2, el elemento está al principio

elemento = 10



priE
actual
ant

Cómo me doy cuenta que está al principio?

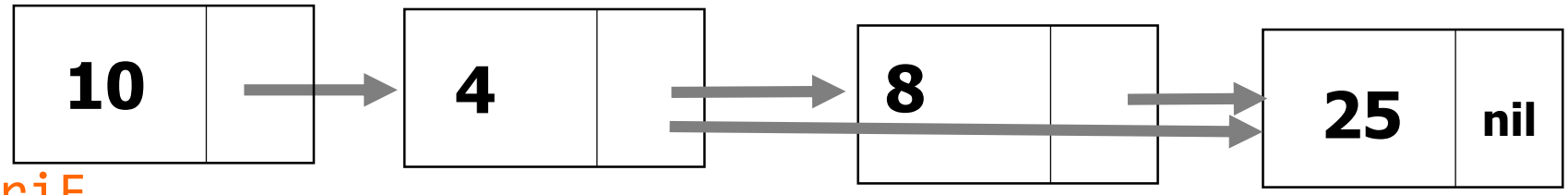
1.Preparo los punteros para el recorrido.
2.Reorganizo punteros, incluyendo al puntero inicial.
3.Realizo el dispose.

CADP – Estructura de Datos – Lista – Operaciones

Eliminar un elemento.

Existen 4 casos: CASO 3 , el elemento NO está al principio

```
elemento = 8
```



priE
actual
ant

Cómo me doy cuenta que no está al principio?

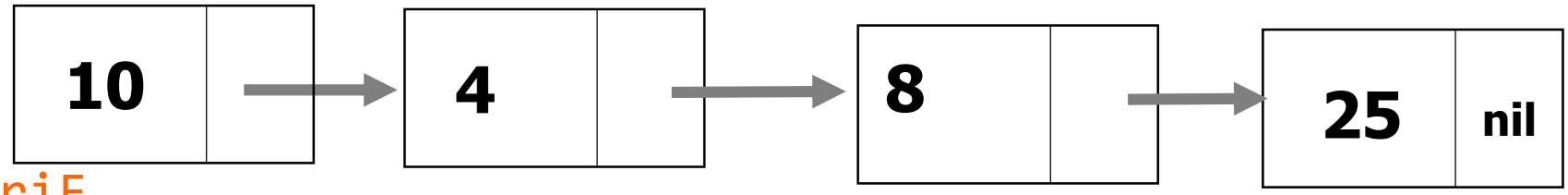
1. Preparo los punteros para el recorrido.
2. Reorganizo punteros
3. Realizo el dispose.

CADP – Estructura de Datos – Lista – Operaciones

Eliminar un elemento.

Existen 4 casos: CASO 4 , el elemento NO está en la lista

elemento = 12



priE
actual
ant

Cómo me doy cuenta que no está?

1.Preparo los punteros para el recorrido.
2.Como el elemento no está no hago nada.

CADP – Estructura de Datos – Lista – Operaciones

```
Procedure eliminar(var p:listaE valor:integer);
Var
 act,ant:listaE;

Begin
  act:=p;
  while (act <> nil) and (act^.elem <> valor) do begin
    ant:=act; act:= act^.sig;
  end;
  if (act <> nil) then
   if (act = p) then begin
     p:= p^.sig; dispose (act);
   end
   else begin
    ant^.sig:= act^.sig;
    dispose (act);
  end;
End;
```

Es el primero

NO es el primero

una solución mejor?

CADP – Estructura de Datos – Lista – Operaciones

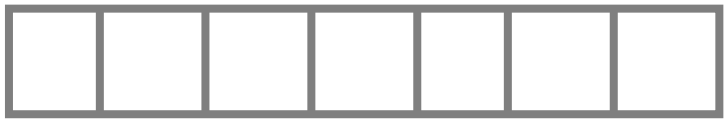
```
Procedure eliminar(var p:listaE valor:integer);
Var
 act,ant:listaE;

Begin
  while (act <> nil) and (act^.elem <> valor) do begin
    ant:=act;
    act:= act^.sig;
  end;

  if (act <> nil) then begin
   if (act = p) then p:= p^.sig;

  else
    ant^.sig:= act^.sig;
  dispose (act);
 end;
End;
```

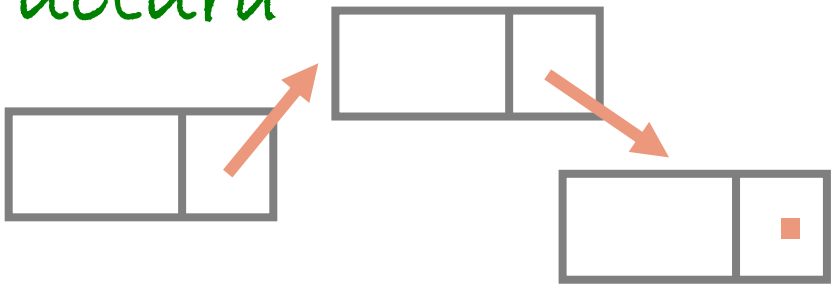
Que ocurre si aparece más de una vez?

CADP – Estructura de Datos – Lista – Comparación

CADP – Estructura de Datos – Lista – Comparación

Características de la estructura





Estática

Homogénea

Indexada

Dinámica

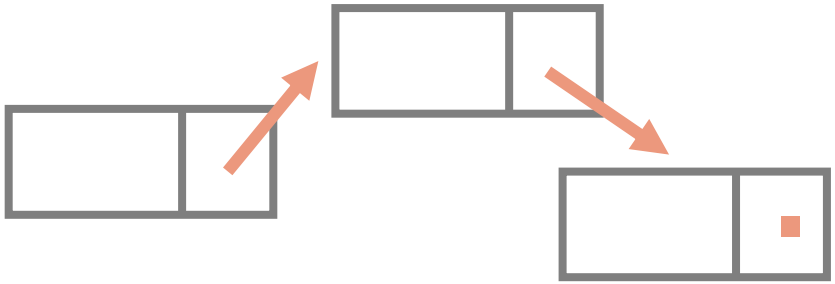
Homogénea

Lineal

CADP – Estructura de Datos – Lista – Comparación

Almacenamiento en memoria

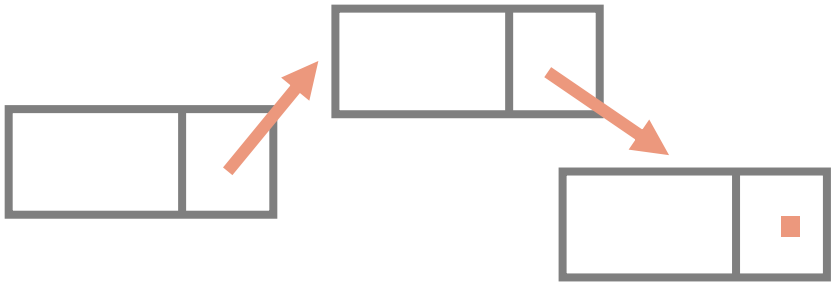
Al ser el arreglo una estructura estática, el mismo se encuentra alojado en posiciones de memoria consecutivas



Al ser la lista una estructura dinámica, cada vez que se necesita agregar un elemento se solicita memoria, por lo tanto no existe relación entre las posiciones de memoria de los elementos

CADP – Estructura de Datos – Lista – Comparación

Operación de agregar



Adelante:
Se debe hacer un corrimiento para generar el espacio.

Incrementar la dimensión lógica.

Puede no poder hacerse la operación.

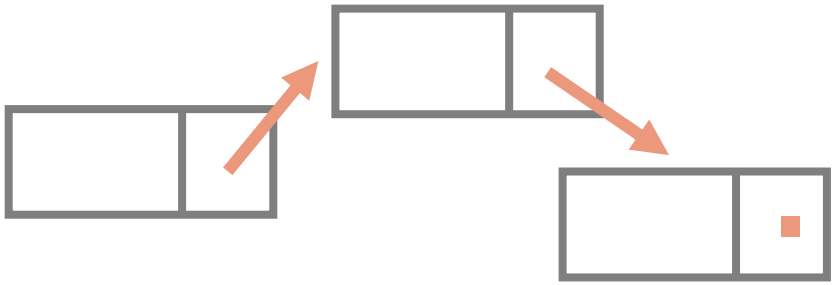
Adelante:
Se debe solicitar un espacio.

Reacomodar los punteros.

Siempre puede hacerse la operación.

CADP – Estructura de Datos – Lista – Comparación

Operación de agregar



Al final:
Se agrega el elemento en la dimensión lógica + 1.

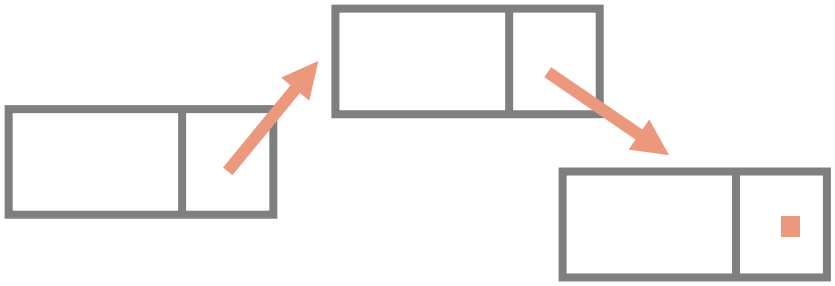
Incrementar la dimensión lógica.

Puede no poder hacerse la operación.

Al final:
Existen dos alternativas:
-Recorrer toda la lista, solicitar espacio y reacomodar los punteros.
-Llevar un puntero que contenga la dirección del último elemento de la lista, solicitar espacio y reacomodar los punteros.
Siempre puede hacerse la operación.

CADP – Estructura de Datos – Lista – Comparación

Operación de insertar



Debe buscarse la posición.
Una vez encontrada se realiza un corrimiento para hacer lugar.
Se carga el elemento.
Se incrementa la dimensión lógica.

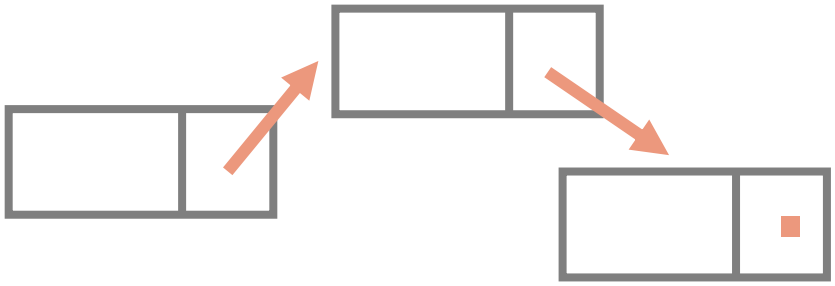
Puede ser que no se pueda hacer la operación.

Solicitar espacio para el nuevo dato.
Debe buscarse la posición.
Una vez encontrada se reorganizan los punteros.

Siempre puede hacerce la operación.

CADP – Estructura de Datos – Lista – Comparación

Operación de eliminar



Debe buscarse el elemento.

Una vez que el elemento es encontrado, realizar un corrimiento desde la posición donde esta el elemento hasta el final.

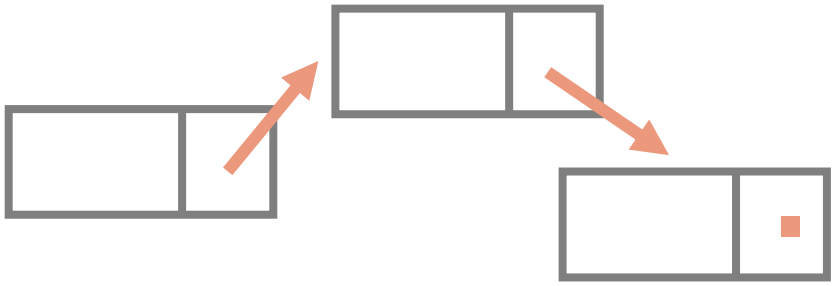
Decrementar la dimensión lógica

Debe buscarse el elemento.

Una vez que el elemento es encontrado, se reorganizan los punteros.

CADP – Estructura de Datos – Lista – Comparación

Acceso a la estructura

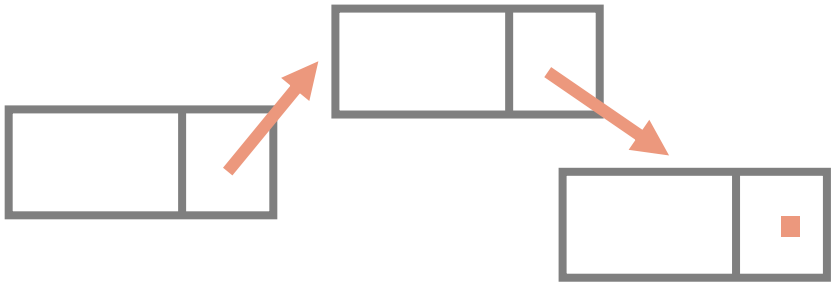


Al ser una estructura indexada, para acceder a la posición **I** (si es válida), se utiliza el acceso directo.

Al ser una estructura lineal, para acceder a la posición **I**, se debe pasar por los **I-1** elementos anteriores.

CADP – Estructura de Datos – Lista – Comparación

Memoria ocupada



Al ser una estructura estática, siempre ocupa la misma cantidad de memoria estática.

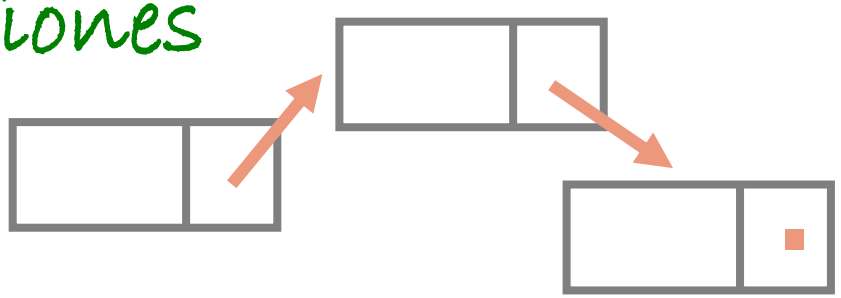
Al ser una estructura dinámica su tamaño cambia al agregar o eliminar elementos.

Ambas estructuras vacías?

Ambas con la misma cantidad de elementos?

CADP – Estructura de Datos – Lista – Comparación

Parámetros en las operaciones



Cuando un arreglo que es pasado por valor Por valor se copia todo el arreglo en el parámetro correspondiente.

Cuando es pasado por referencia se pasa sólo la dirección.

Cuando una lista es pasada por valor se copia la dirección inicial de la lista.

Cuando es pasada por referencia se pasa la dirección inicial de la lista.

Qué tipo de parámetro sería una lista de enteros a la que quiero multiplicar por 2, cada elemento?