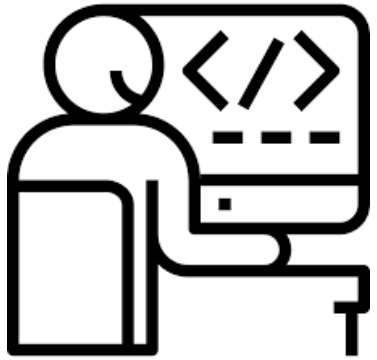


Conceptos de Algoritmos Datos y Programas

CADP – TIPOS DE DATOS - LISTA

Creación de una lista.

Agregar nodos al comienzo de la lista.

Recorrido de una lista.

Agregar nodos al final de la lista.

Insertar nodos en una lista ordenada

Eliminar nodos de una lista

CADP – **TEMAS**

- Estructura de Datos - LISTA
- Operación de CREACION
- Operación de RECORRIDO

CREAR UNA LISTA

Implica marcar que la lista no tiene una dirección inicial de comienzo.

Qué valor se le asigna a un puntero para indicar que no tiene una dirección asignada?

```
Program uno;

Type listaE= ^datosEnteros;

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

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

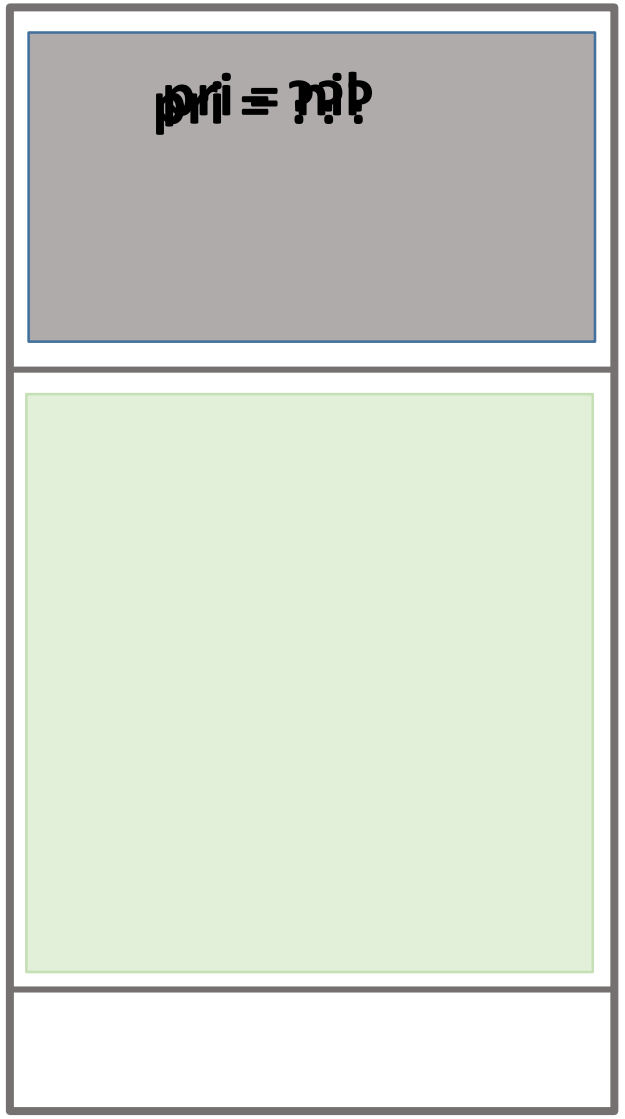
CADP – TIPOS DE DATOS - LISTA

CREAR

```
Program uno;

Type listaE= ^datosEnteros;

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


Var
  pri: listaE;


Begin
  pri:=nil;
End.
```

Por qué no se hace new (pri)?

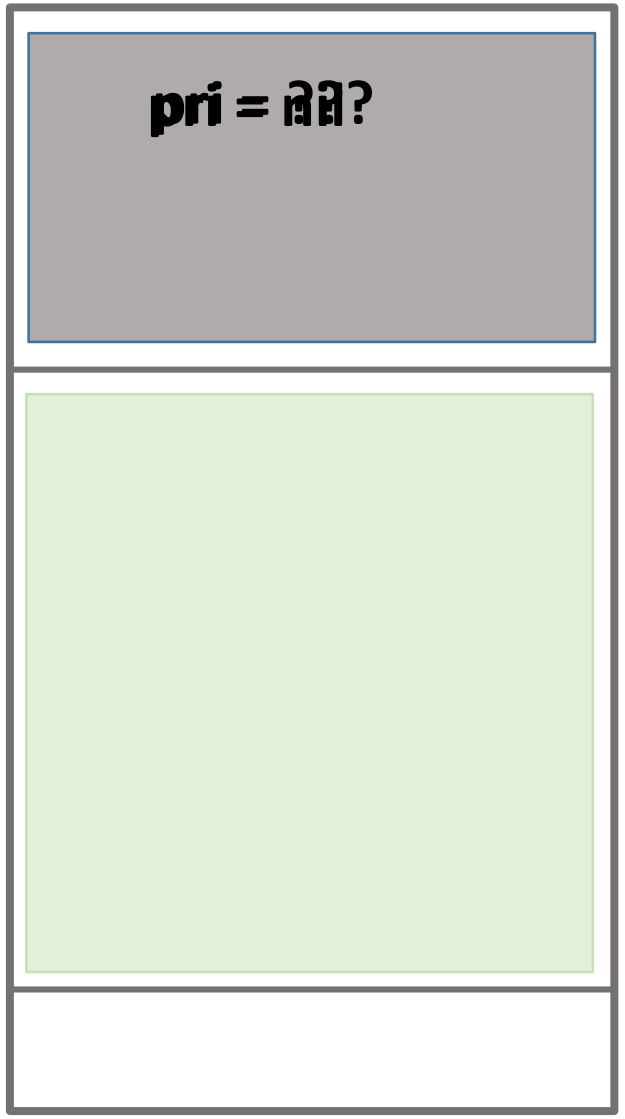
Se puede modularizar el crear?



CADP – TIPOS DE DATOS - LISTA

CREAR

```
Program uno;

Type listaE= ^datosEnteros;

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

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

Var
  pri: listaE;

Begin
  crear (pri);
End.
```



RECORRER UNA LISTA

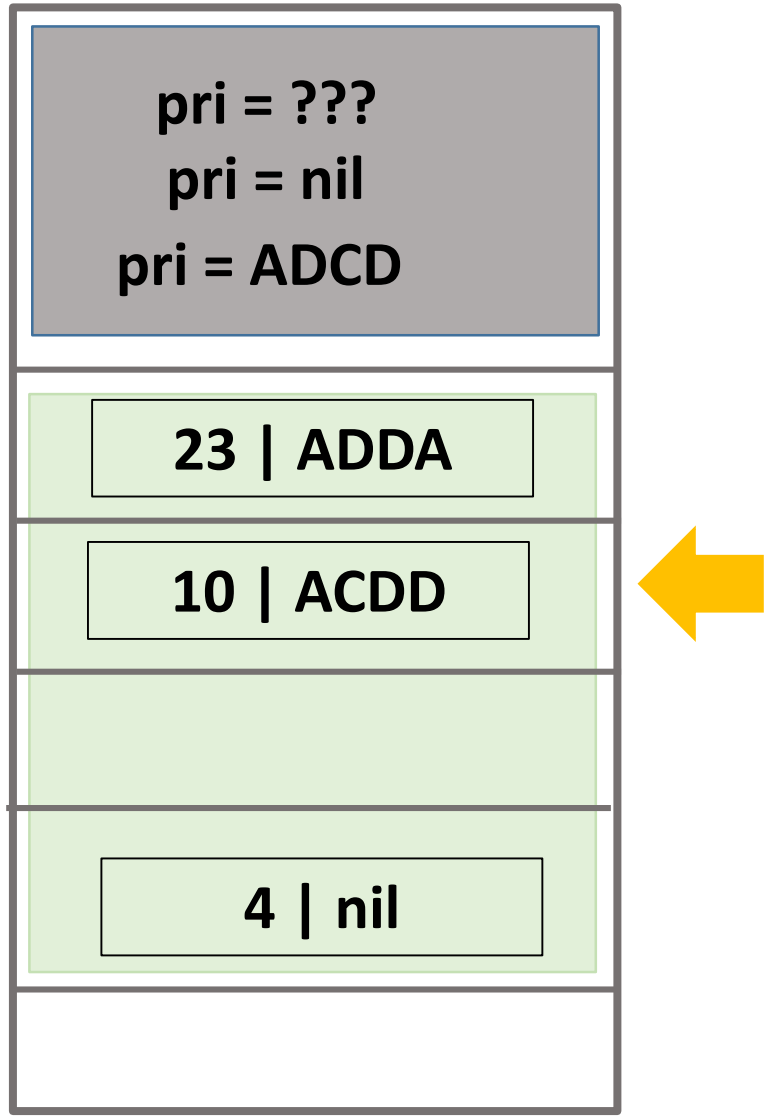
Implica posicionarse al comienzo de la lista y a partir de allí ir "pasando" por cada elemento de la misma hasta llegar al final.

```
Program uno;

Type listaE= ^datosEnteros;

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

Var
  pri: listaE;
```

CADP – TIPOS DE DATOS - LISTA

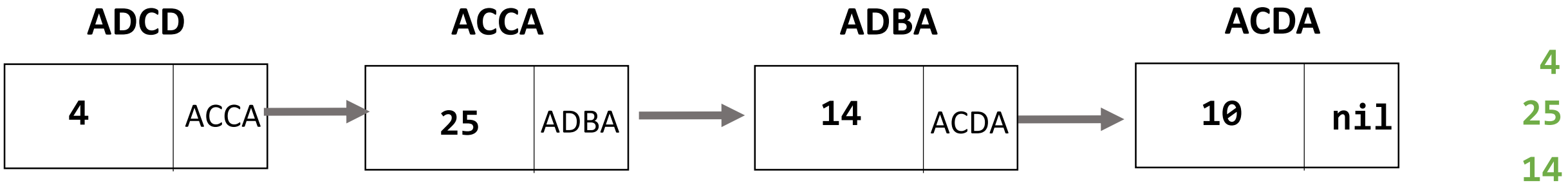
RECORRER UNA LISTA

```
Program uno;

Type listaE= ^datosEnteros;

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

Var
  pri: listaE;

Begin
  crear (pri);
  cargarLista (pri); //Lo implementaremos más adelante

  recorrerLista (pri);
 End.
```



CADP – TIPOS DE DATOS - LISTA

RECORRER UNA LISTA

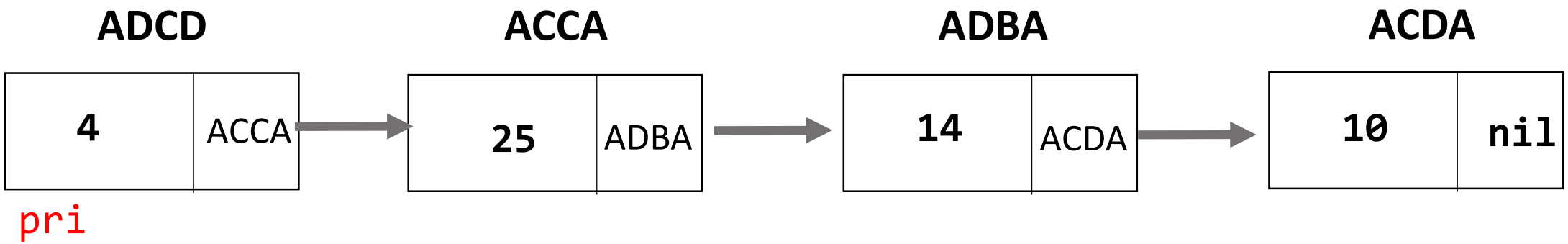


Inicializo una variable **auxiliar** con la dirección del puntero inicial de la lista

mientras (no sea el final de la lista)
    proceso el elemento (ej: imprimo, sumo, modifico)
    avanzo al siguiente elemento de **auxiliar**

CADP – TIPOS DE DATOS - LISTA

RECORRER UNA LISTA




```
procedure recorrerLista (pI: listaE);
Var
 aux:listaE;

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

Es correcto?

Si la lista está **vacía**, (aux^.sig) da error.

Si la lista tiene **un solo elemento** (aux^.sig <> nil) da falso.

Si la lista tiene **muchos elementos** no imprime el último

CADP – TIPOS DE DATOS - LISTA

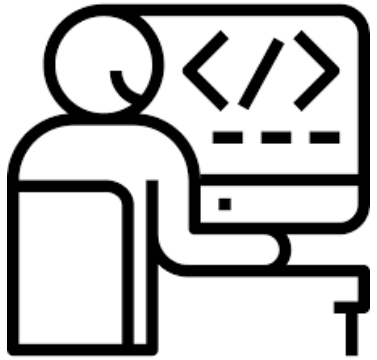
RECORRER UNA LISTA

```
procedure recorrerLista (pI: listaE);
Var
 aux:listaE;

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

Es necesaria la variable aux?

ALTERNATIVA

```
procedure recorrerLista (pI: listaE);

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



Conceptos de Algoritmos Datos y Programas

CADP – TIPOS DE DATOS - LISTA

Creación de una lista.

Agregar nodos al comienzo de la lista.

Recorrido de una lista.

Agregar nodos al final de la lista.

Insertar nodos en una lista ordenada

Eliminar nodos de una lista

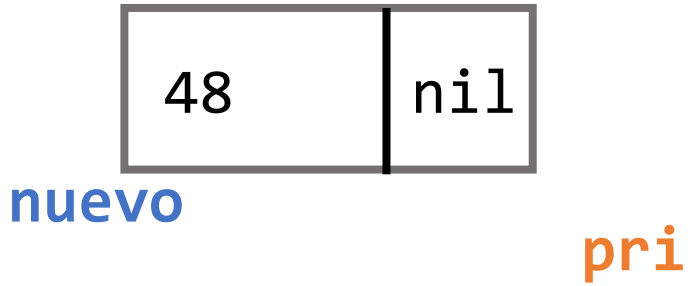
CADP – **TEMAS**

- Operación de AGREGAR ADELANTE
- Operación de AGREGAR AL FINAL

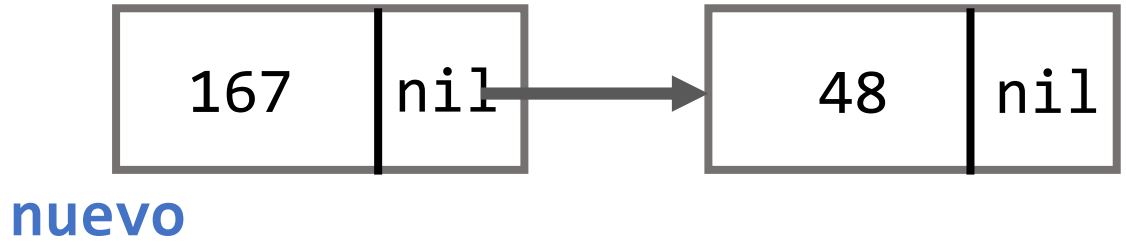
AGREGAR ADELANTE

Implica generar un nuevo nodo y agregarlo como primer elemento de la lista.

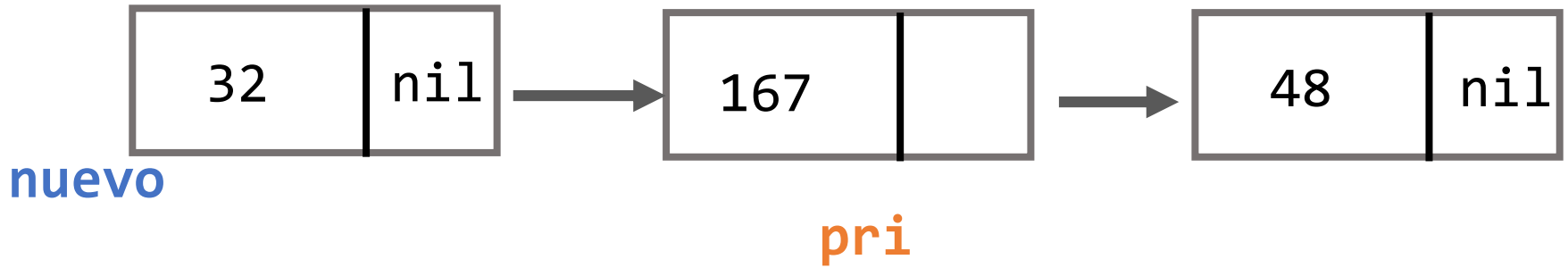
- `pri = nil`



- `pri <> nil`



Cómo lo escribo?



CADP – TIPOS DE DATOS - LISTA AGREGAR ADELANTE

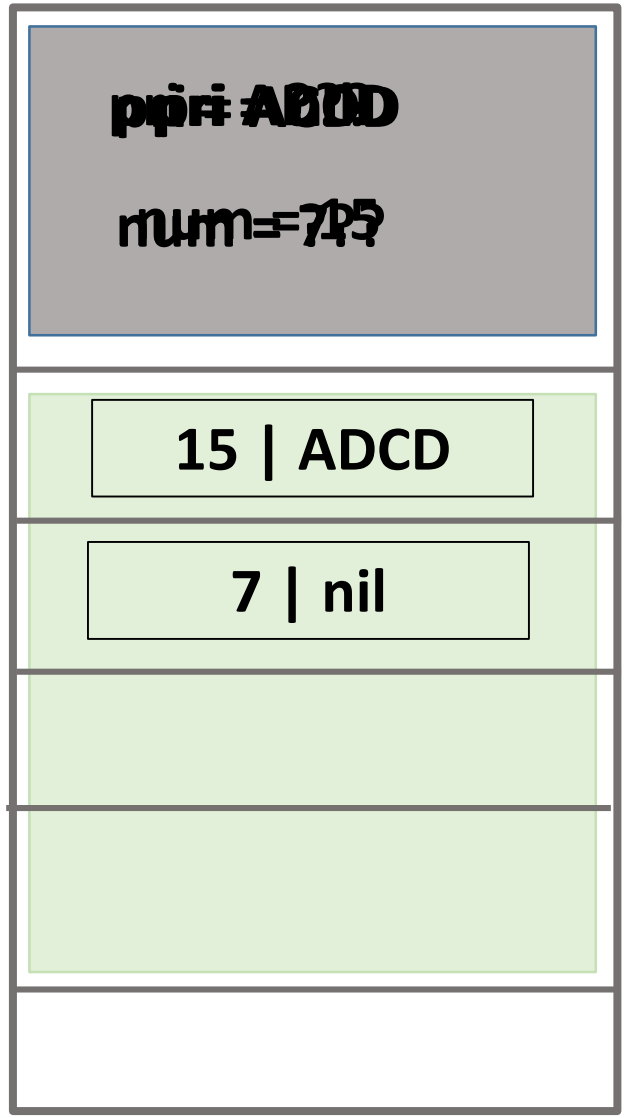
Implica generar un nuevo nodo y agregarlo como primer elemento de la lista.

```
Reservo espacio en memoria nuevo elemento.

si (es el primer elemento a agregar)
  asigno al puntero inicial la dirección del nuevo elemento.

sino
  indico que el siguiente de nuevo elemento es el puntero inicial.
  actualizo el puntero inicial de la lista con la dirección del nuevo
  elemento.
```

CADP – TIPOS DE DATOS - LISTA

AGREGAR ADELANTE

```
Program uno;

Type listaE= ^datosEnteros;

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

Var
  pri: listaE;
  num:integer;
Begin
  crear (pri);
  read (num);
  agregarAdelante (pri,num);
  read (num);
  agregarAdelante (pri,num);
```



CADP – TIPOS DE DATOS - LISTA

AGREGAR ADELANTE

```
procedure agregarAdelante (var pI:listaE; num:integer);
Var
 nuevo:listaE;

Begin
   new (nuevo); nuevo^.elem:= num; nuevo^.sig:=nil;

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

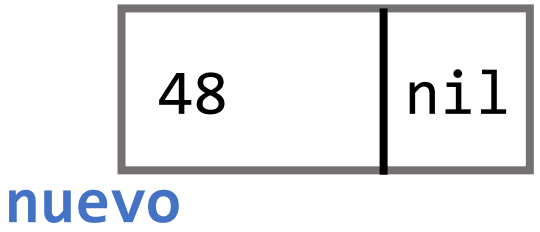
Creo espacio para el nuevo elemento

Evalúo el caso y reasigno los punteros

AGREGAR AL FINAL

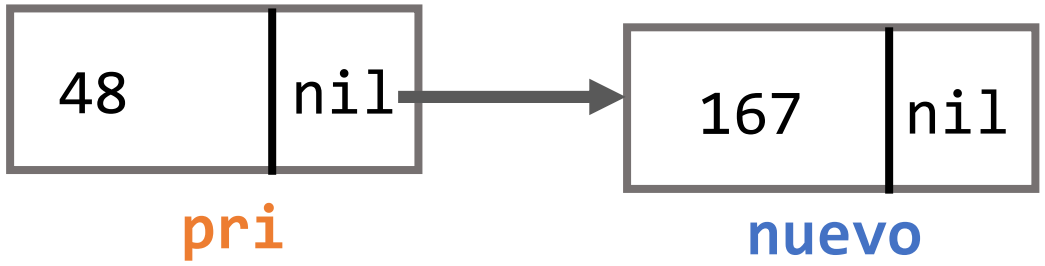
Implica generar un nuevo nodo y agregarlo como último elemento de la lista.

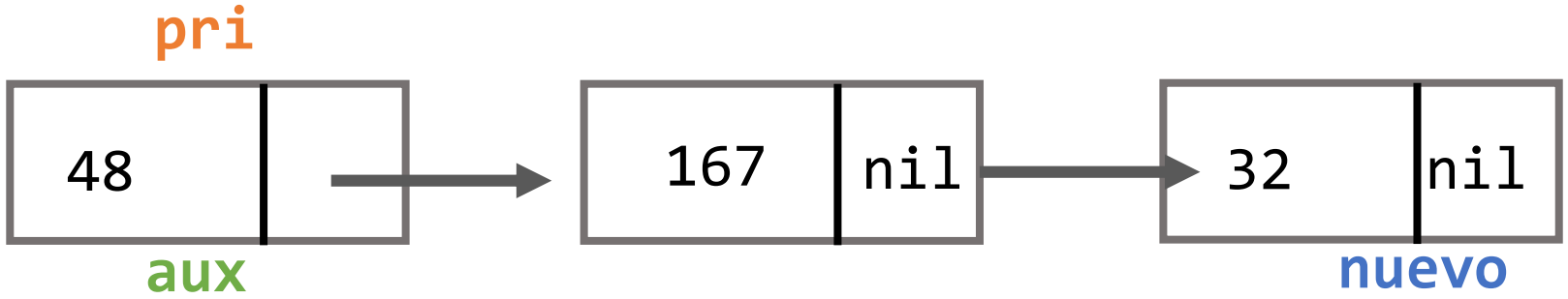
- `pri = nil`



- `pri <> nil`

Cómo lo escribo?





CADP – TIPOS DE DATOS - LISTA AGREGAR AL FINAL

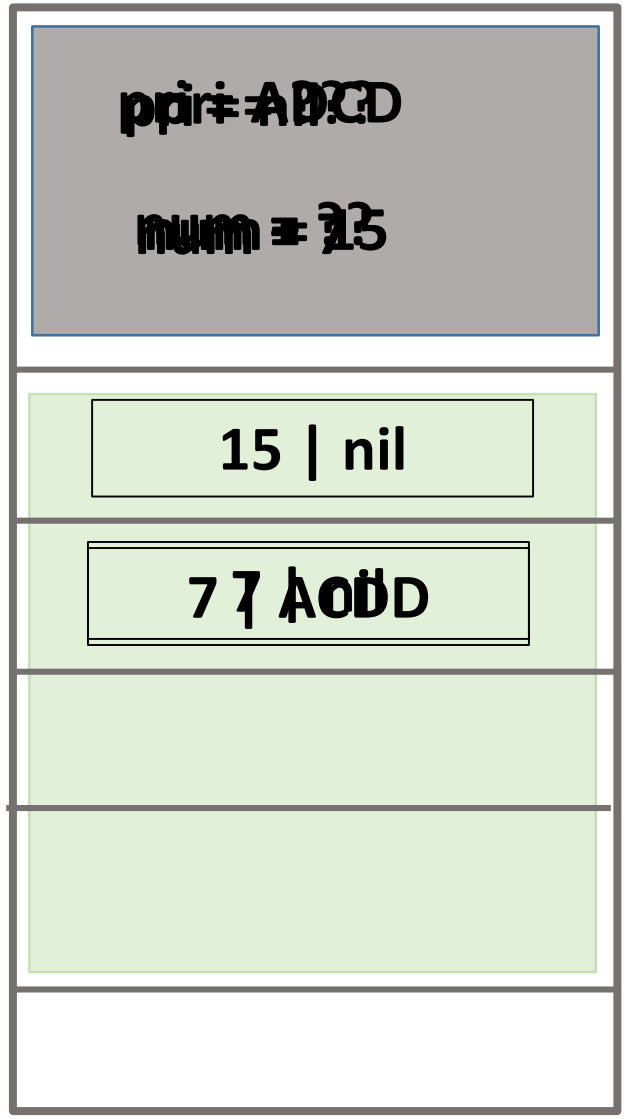
Implica generar un nuevo nodo y agregarlo como último elemento de la lista.

```
Reservo espacio en memoria nuevo elemento.

si (es el primer elemento a agregar)
  asigno al puntero inicial la dirección del nuevo elemento.

sino
  inicializo un puntero auxiliar aux
  mientras (no llegue al último elemento)
   avanzo en la lista.
  actualizo como siguiente del último nodo al nuevo elemento
```

CADP – TIPOS DE DATOS - LISTA

AGREGAR AL FINAL

```
Program uno;

Type listaE= ^datosEnteros;

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

Var
  pri: listaE;
  num:integer;
Begin
  crear (pri);
  read (num);
  agregarAlFinal (pri,num);
  read (num);
  agregarAlFinal (pri,num);
```



CADP – TIPOS DE DATOS - LISTA

AGREGAR AL FINAL

```
procedure agregarAlFinal (var pI:listaE; num:integer);
Var
 nuevo,aux:listaE;

Begin
   new (nuevo); nuevo^.elem:= num; nuevo^.sig:=nil;

    if (pI = nil) then pI:= nuevo
    else begin
         aux:= pI;
         while (aux ^.sig <> nil) do
           aux:= aux^.sig;

         aux^.sig:=nuevo;
        end;
 End;
```

Si agrego al final por qué paso por referencia el puntero inicial?

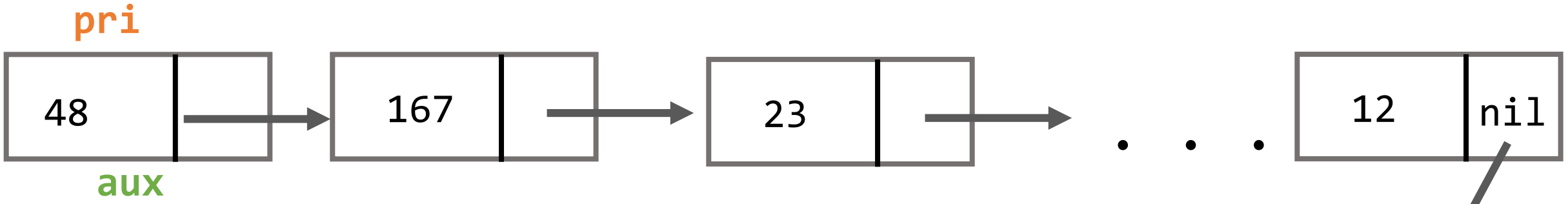
Por qué en la condición del while se pregunta por el aux^.sig?

Evalúo si la lista está vacía

Recorro y quedo parado en el último elemento

Le indico al último que ahora su siguiente es nuevo

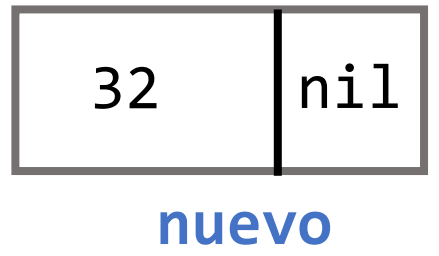
AGREGAR AL FINAL EN UNA LISTA (OPCION 2)



Cada vez que debo agregar un nuevo elemento al final de la lista, se tiene que recorrer la misma de manera completa hasta llegar al final.

SE PUEDE MEJORAR?

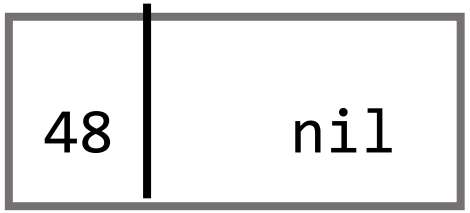
Cómo lo escribo?



CADP – TIPOS DE DATOS - LISTA

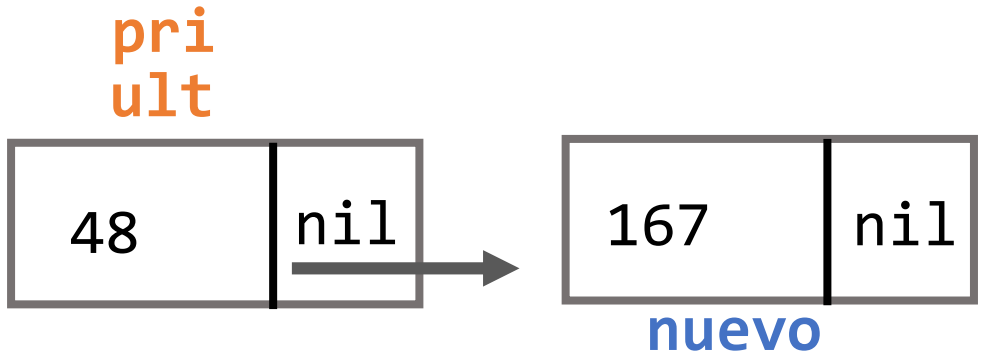
AGREGAR AL FINAL – 2

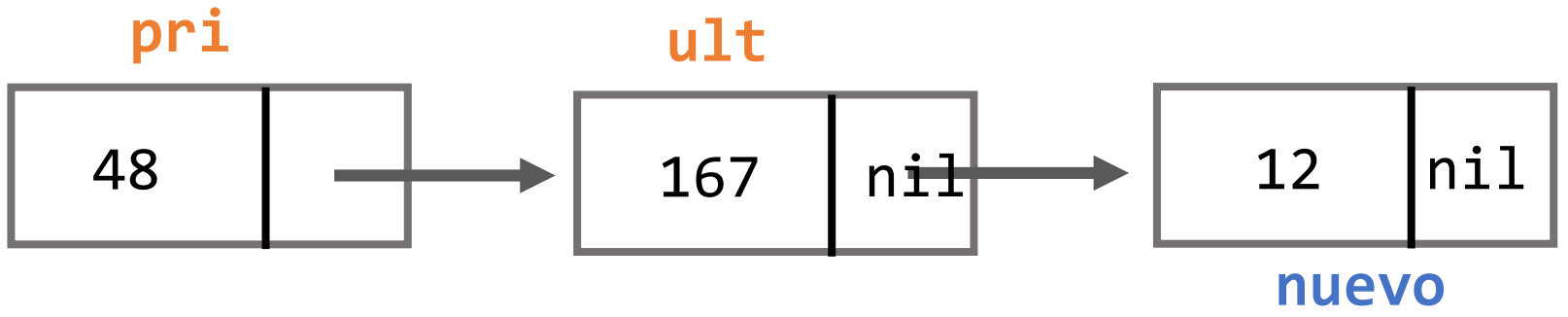
- pri = nil



pri
ult

- pri <> nil





Cómo lo escribo?

AGREGAR AL FINAL EN UNA LISTA (OPCION 2)

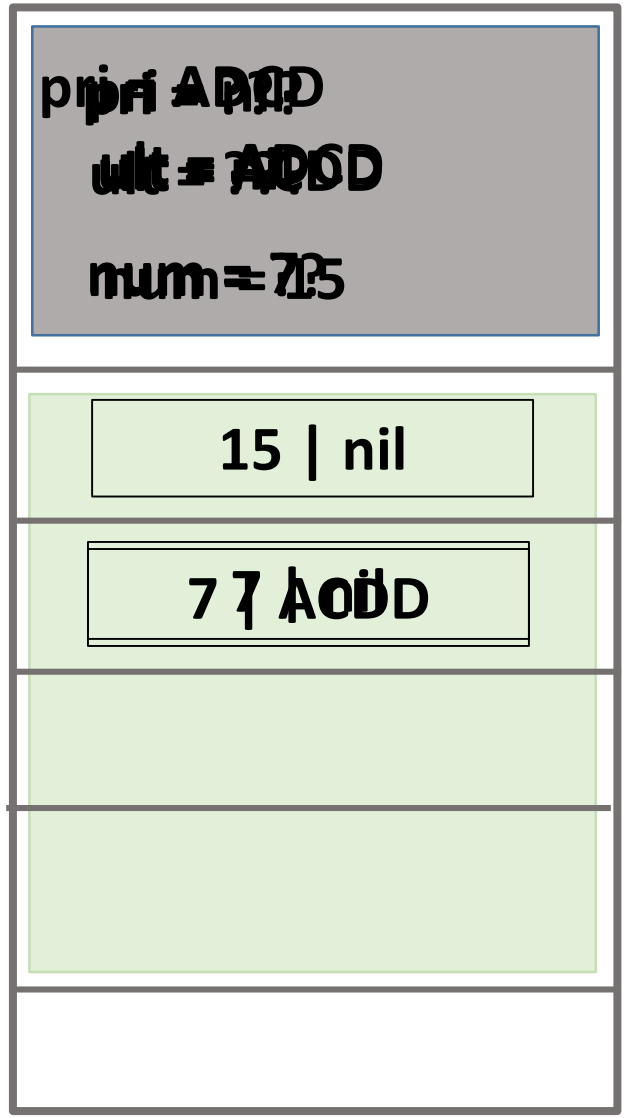
Implica generar un nuevo nodo y agregarlo como último elemento de la lista.

```
Reservo espacio en memoria nuevo elemento.

si (es el primer elemento a agregar)
  asigno al puntero inicial la dirección del nuevo elemento.
  asigno al puntero final la dirección del nuevo elemento.

sino
  actualizo como siguiente del puntero final al nuevo elemento
  actualizo el la dirección del puntero final
```

CADP – TIPOS DE DATOS - LISTA

AGREGAR AL FINAL -2

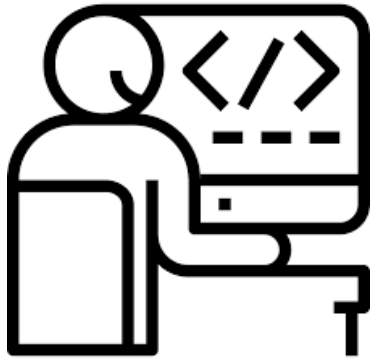
```
Program uno;

Type listaE= ^datosEnteros;

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

Var
  pri,ult: listaE;
  num:integer;
Begin
  crear (pri,ult);
  read (num);
  agregarAlFinal2 (pri,ult,num);
  read (num);
  agregarAlFinal2 (pri,,ult,num);
```



CADP – TIPOS DE DATOS - LISTA

AGREGAR AL FINAL-2

```
procedure agregarAlFinal2 (var pI,pU:listaE; num:integer);
Var
 nuevo:listaE;

Begin
   new (nuevo); nuevo^.elem:= num; nuevo^.sig:=nil;
   if (pI = nil) then begin
      pI:= nuevo;                    <----- Evalúo si la lista está vacía
      pU:= nuevo;
   end
   else begin
         pU^.sig:=nuevo;
         pU:= nuevo;          <----- Actualizo el siguiente del
        end;                          último nodo y al último nodo
End;
```



Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Estructura de Datos - LISTA
- Características de una LISTA
- Operaciones de una LISTA

CADP – TIPOS DE DATOS - LISTA

Realizar un programa que lea números que representan edades de personas hasta leer la edad -1. Finalizada la lectura se quiere informar cual fue la edad máxima leída.

```
10
4
57
62
39
-1
```

Dónde almaceno las edades?

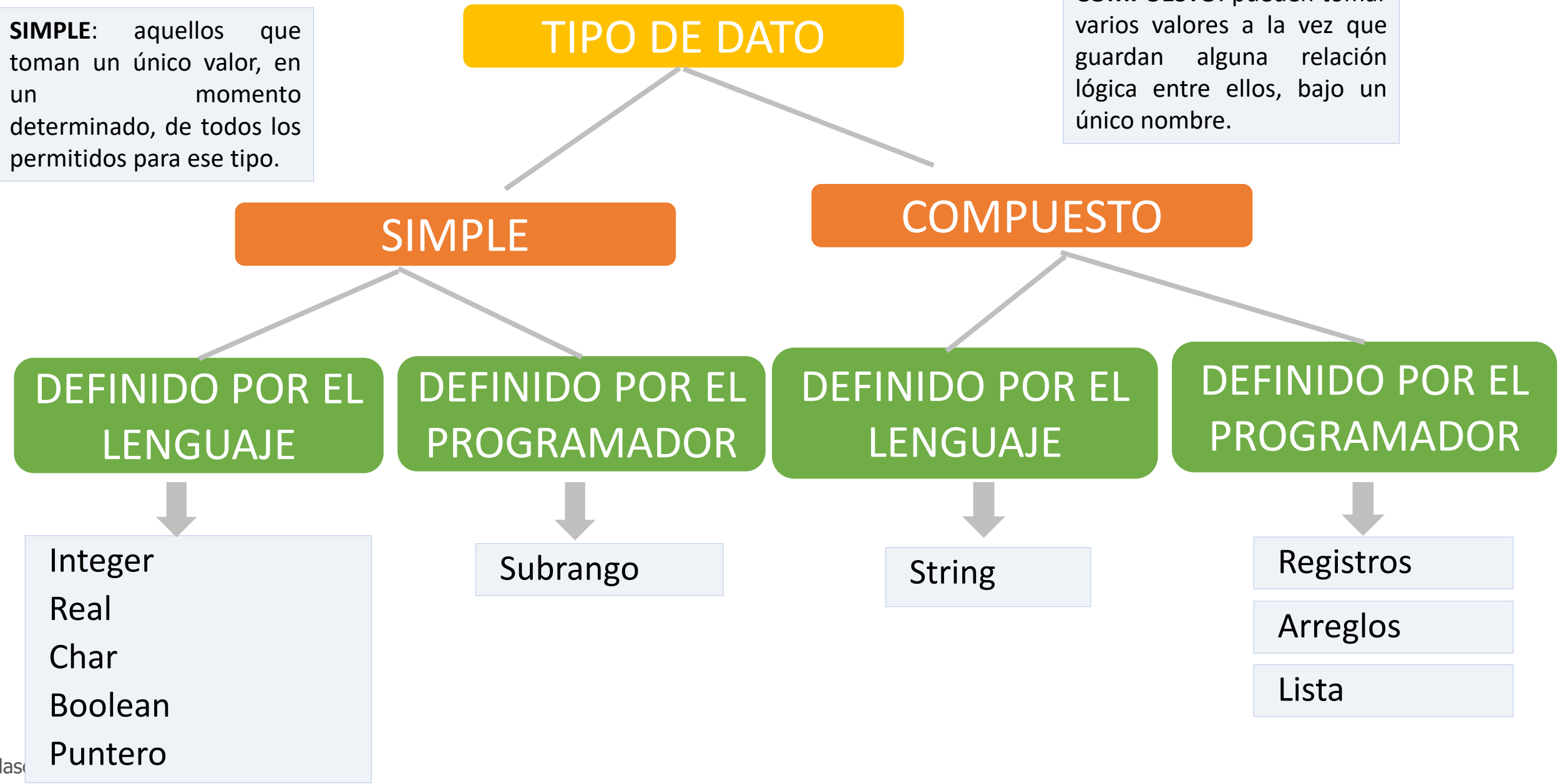
Necesito una estructura que pueda ir agregando datos y por lo tanto su tamaño pueda ir variando en la ejecución del programa (estructura dinámica)

LISTA

CADP – TIPOS DE DATOS - LISTA

SIMPLE: aquellos que toman un único valor, en un momento determinado, de todos los permitidos para ese tipo.

COMPUESTO: pueden tomar varios valores a la vez que guardan alguna relación lógica entre ellos, bajo un único nombre.



CADP – TIPOS DE DATOS - LISTA

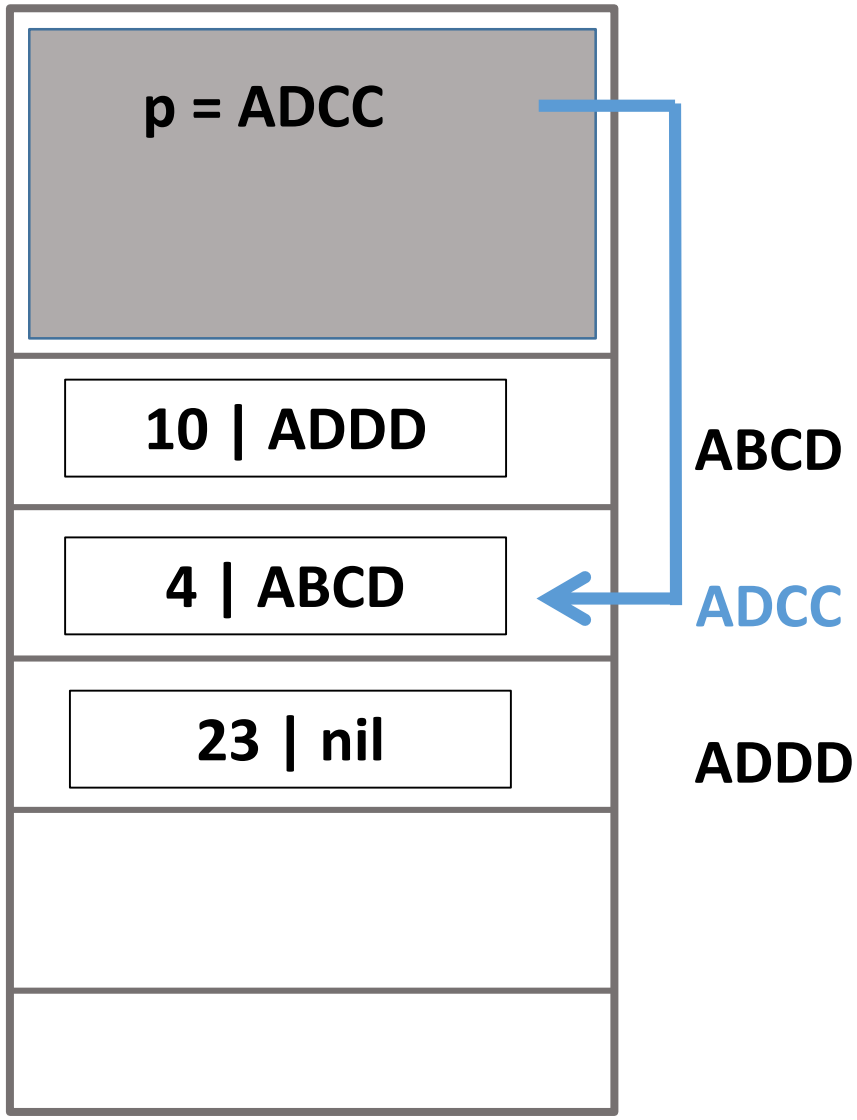
Es una colección de nodos. Cada nodo contiene un elemento (valor que se quiere almacenar en la lista) y una dirección de memoria dinámica que indica donde se encuentra el siguiente nodo de la lista.

Toda lista tiene un nodo inicial.

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

Gráficamente …

CADP – TIPOS DE DATOS - LISTA

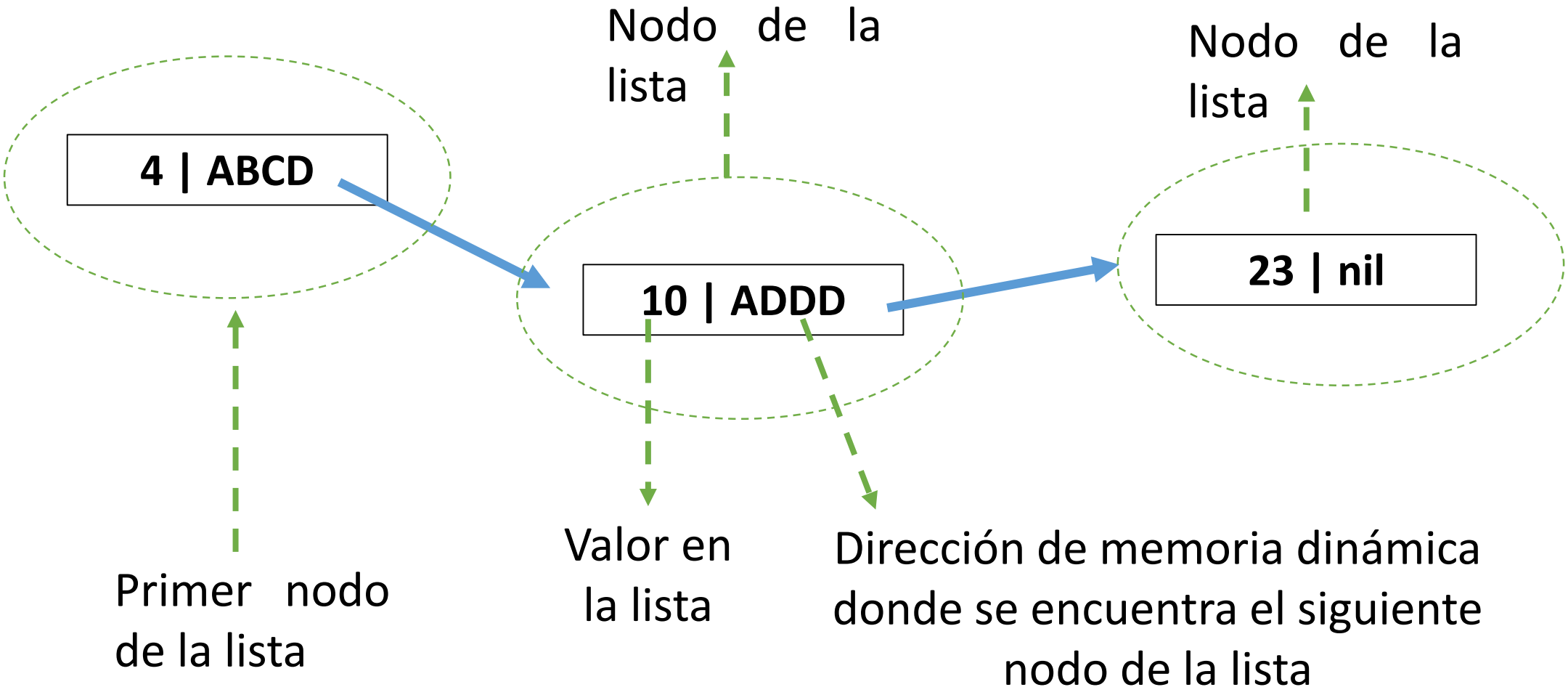


En **memoria estática** se declara una variable tipo PUNTERO (ya que son las única que pueden almacenar direcciones). La dirección almacenada en esa variable representa la dirección donde comienza la lista. Inicialmente ese puntero no contiene ninguna dirección.

Luego a medida que se quiere agregar elementos a la lista (nodo), se reserva una dirección de **memoria dinámica** y se carga el valor que se quiere guardar.

El último nodo de la lista indica que la dirección que le sigue es nil.

CADP – TIPOS DE DATOS - LISTA



CADP – TIPOS DE DATOS - LISTA

CARACTERISTICAS

Cómo se declara?

HOMOGENEA → Los elementos pueden son del mismo tipo .

DINAMICA → El tamaño puede cambiar durante la ejecución del programa.

LINEAL → Cada nodo de la lista tiene un nodo que lo sigue (salvo el último) y uno que lo antecede (salvo el primero).

SECUENCIAL → El acceso a cada elemento es de manera secuencial, es decir, para acceder al elemento 5 (por ejemplo) debo pasar por los 4 anteriores.

Cada vez que se necesite agregar un nodo se deberá reservar memoria dinámica (new) y cuando se quiera eliminar un nodo se debe liberar la memoria dinámica (dispose) .

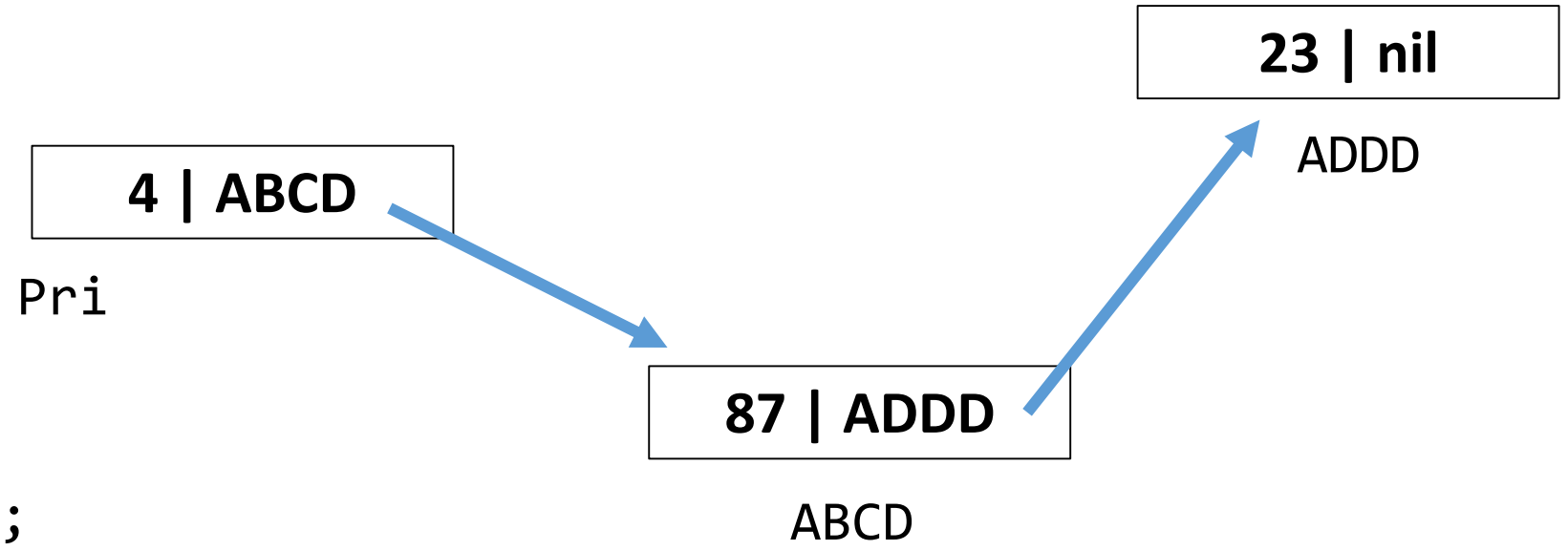
CADP – TIPOS DE DATOS - LISTA DECLARACION

```
Program uno;

Type
  nombreTipo= ^nombreNodo;

  nombreNodo = record
       elemento: tipoElemento;
       punteroSig: nombreTipo;
  end;


Var
  Pri: nombreTipo;
```

tipoElemento es cualquiera de los tipos vistos (entero,char,boolean,registro,arreglo,real,subrangol).
Es una estructura recursiva.
El orden de la declaración debe respetarse

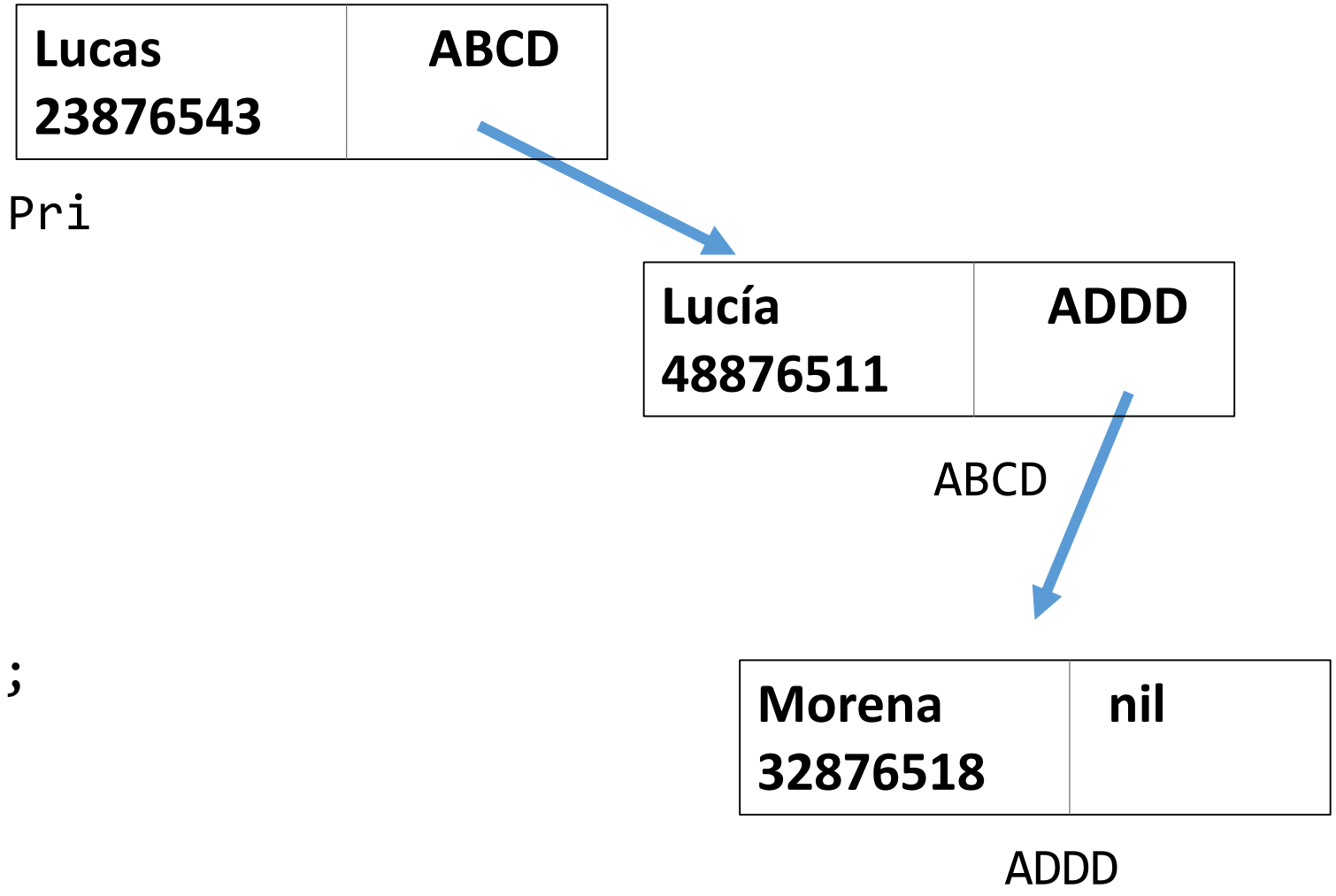
CADP – TIPOS DE DATOS - LISTA DECLARACION

```
Program uno;

Type
  listaE= ^nodo;

  nodo = record
       elemento: integer;
       punteroSig: listaE;
  end;


Var
  Pri: listaE;
```



CADP – TIPOS DE DATOS - LISTA DECLARACION

```
Program dos;

Type
  persona = record
    nom:string;
    dni:integer;
  end;

  listaE= ^nodo;
  nodo = record
       elemento: persona;
       punteroSig: listaE;
  end;

Var
  Pri: listaE;
```



CADP – TIPOS DE DATOS - LISTA

Creación de una lista.

Agregar nodos al comienzo de la lista.

Recorrido de una lista.

Agregar nodos al final de la lista.

Insertar nodos en una lista ordenada

Eliminar nodos de una lista