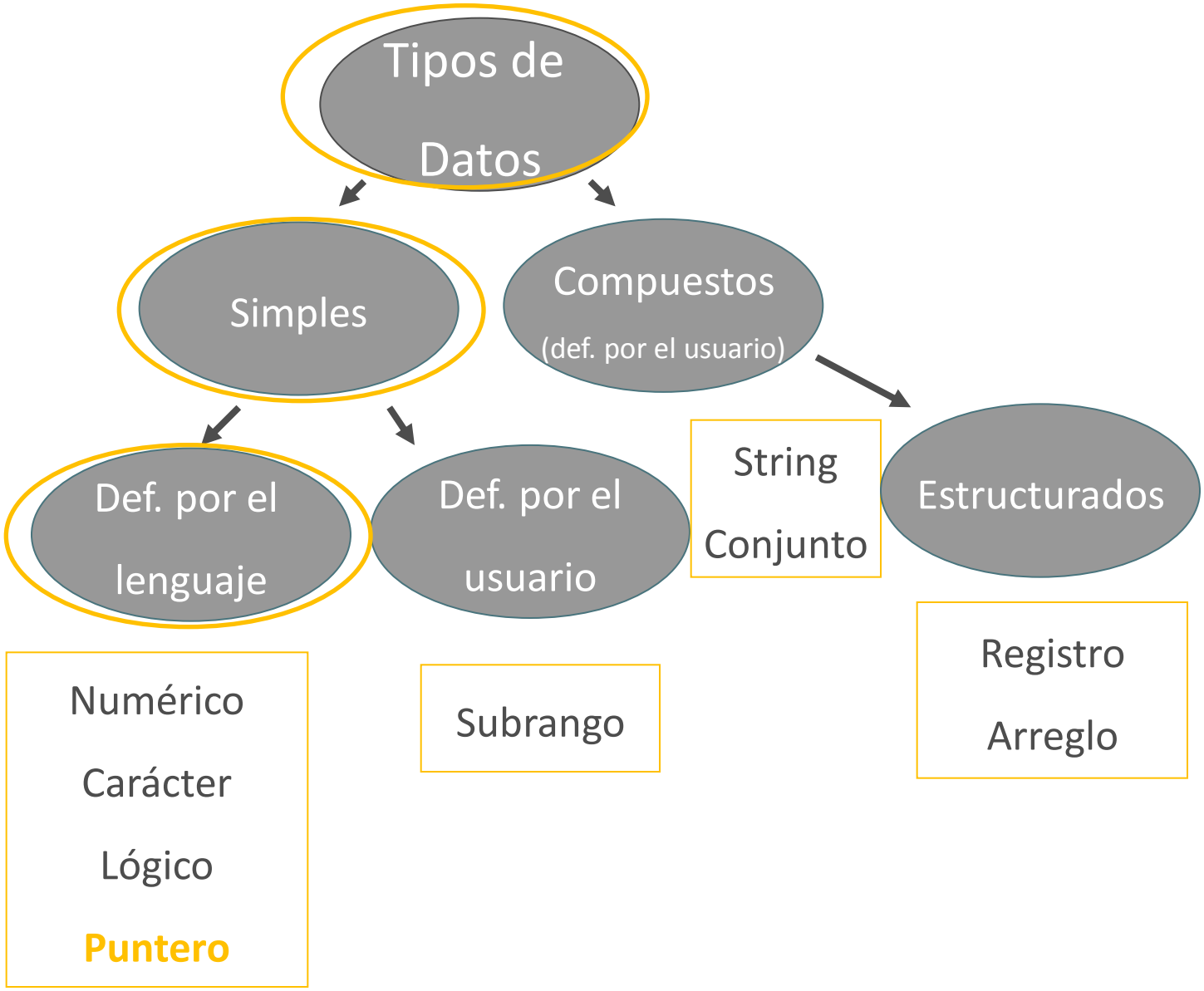
Programación I

TEMAS de la CLASE

1 Tipo de dato PUNTERO

- Características
- Ejercitación

PUNTEROS: Recordemos clasificación...



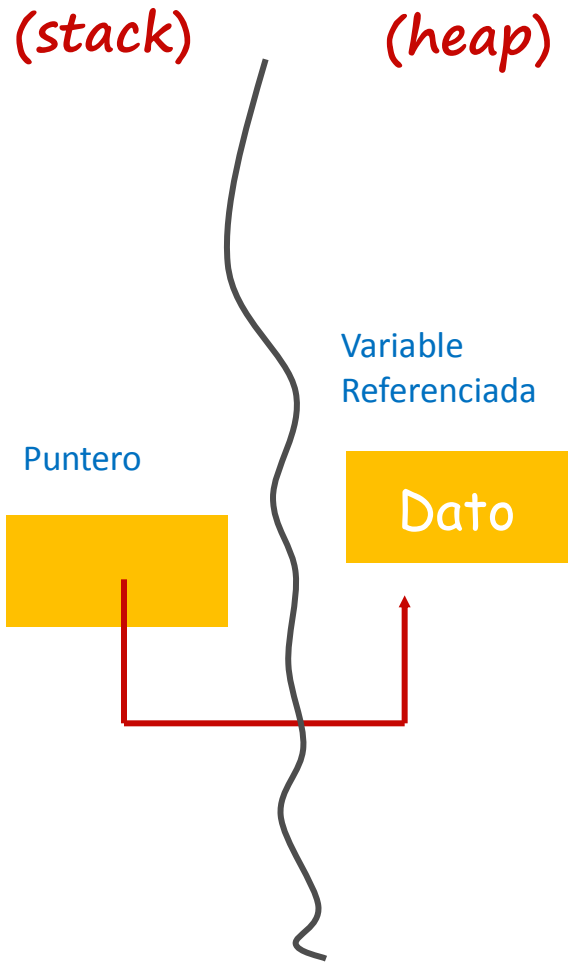
TIPO DE DATO PUNTERO

Un puntero es un tipo de dato simple que contiene la dirección de otro dato.

- Los punteros (en stack) pueden apuntar solamente a variables dinámicas, es decir, a datos que están almacenados en memoria dinámica (heap).

- Una variable de tipo puntero ocupa 4 bytes de memoria (stack) para su representación interna en Pascal.

- Cada variable de tipo puntero puede apuntar a un único tipo de dato (en Heap).



TIPO DE DATO PUNTERO: DECLARACION

- En Pascal, un tipo de dato puntero se define:

TYPE TipoPuntero= ^TipoVariableApuntada;

- Por ejemplo para definir un tipo puntero a un producto:

```
TYPE   producto = record
                    codigo: integer;
                    stock: integer;
                    precio: real;
                  end;

PunteroAProducto = ^producto;
```

Y luego se puede declarar una variable:

VAR Puntero : PunteroAProducto;

- Un dato de tipo puntero puede apuntar a una variable de cualquier tipo.

TIPO DE DATO PUNTERO: EJEMPLOS DE DECLARACION

```
{declaración de tipos}
type
TipoCadena = array [1..10] of char;
PtrCadena = ^TipoCadena;
PtrReal = ^real;
TipoString = string[20];
PtrString = ^TipoString;
Datos = record
              Nombre: TipoString;
              Apellido: TipoString;
              Edad: integer;
              Altura: real
        End;
PtrDatos = ^datos;
```

```
{declaración de variables}
var
   peso : PtrReal; (o ^real)
    t : PtrString;
    frase : PtrString;
    s : TipoString;
    puntero : PtrCadena;
    p, q : PtrDatos;
```

Analicemos

¿Memoria estática?

¿Memoria dinámica?

TIPO DE DATO PUNTERO: OBSERVACIONES IMPORTANTES

- Una variable de tipo puntero ocupa una cantidad de memoria estática fija (4 bytes), independiente del tipo de dato al que apunta.

- Un dato referenciado o apuntado, como los ejemplos vistos, no tienen memoria asignada, o lo que es lo mismo no existe inicialmente espacio reservado en memoria para este dato.

- Para poder emplear variables dinámicas es necesario usar el tipo de dato PUNTERO que permite referenciar nuevas posiciones de memoria que no han sido declaradas a priori y que se van a crear y destruir en tiempo de ejecución.

TIPO DE DATO PUNTERO: OPERACIONES

- Las variables dinámicas son por definición aquellas que se crean cuando se necesitan y se destruyen cuando ya han cumplido con su cometido.

- En Pascal la creación y destrucción de variables dinámicas se realiza mediante los siguientes procedimientos:

New (puntero)

Dispose (puntero)

TIPO DE DATO PUNTERO: OPERACIONES

- Asignación de un valor a una variable puntero
- Asignación de valor al objeto “referenciado” o “apuntado” por el puntero
- Acceso a la información del objeto “referenciado” o “apuntado” por el puntero
- Operaciones de Entrada / Salida???
- Operaciones de comparación
- Eliminación de un objeto apuntado que no se necesita

TIPO DE DATO PUNTERO - OPERACIONES

- Asignación Nula a una variable puntero

```
Type
 TipoString = string[20];
 Datos = record
           Nombre:TipoString;
           Apellido: TipoString;
           Edad: integer;
            Altura: real
         end;
  PtrDatos = ^datos;

Var
   p, q : PtrDatos;
```

p := Nil;

q := Nil;

TIPO DE DATO PUNTERO - OPERACIONES

- Asignación de un valor a una variable puntero

```
Type
 TipoString = string[20];
 Datos = record
          Nombre: TipoString;     21
          Apellido: TipoString;   21
          Edad: integer;           2
          Altura: real             4
         end;
 PtrDatos = ^datos;
 PtrReal = ^real;
 PtrString = ^TipoString;

Var
 p      : PtrDatos;
 peso   : PtrReal;
 frase  : PtrString;
```

RECORDAR Cada variable puntero ocupa 4 bytes

New (p);

Al ejecutar el New, se reserva espacio para el registro (48 bytes)

New (peso);

Al ejecutar el New, se reserva espacio para el real (4 bytes)

New (frase);

Al ejecutar el New, se reserva espacio para el string (21 bytes)

TIPO DE DATO PUNTERO - OPERACIONES

- Asignación de un valor a una variable puntero

Si se tiene

Se puede hacer

p := q

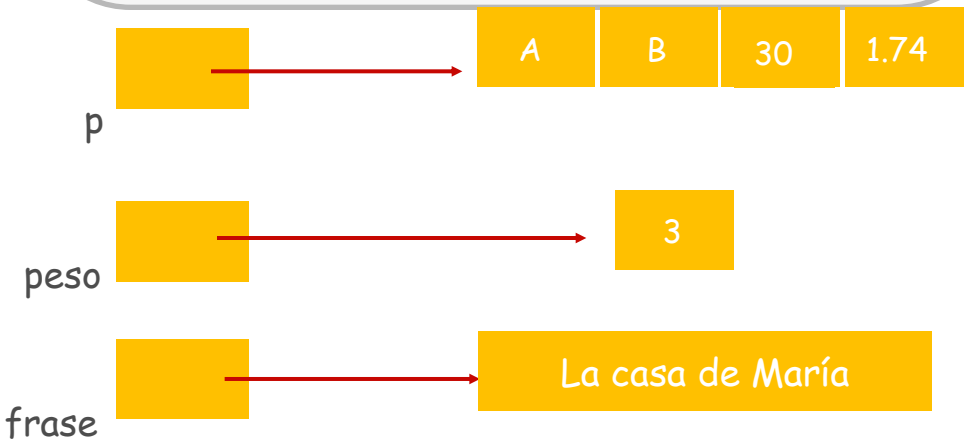
TIPO DE DATO PUNTERO - OPERACIONES

- Asignación de valor a la variable apuntada

```
Type
  TipoString = string[20];
  Datos = record
            Nombre: TipoString;
            Apellido:TipoString;
            Edad: integer;
            Altura: real
          end;
  PtrDatos = ^datos;
  PtrReal = ^real;
  PtrString = ^TipoString;

Var
  p      : PtrDatos;
  peso   : PtrReal;
  frase  : PtrString;
```

```
Begin
New(p); New(peso); New(frase);
read (P^.nombre); read (P^) MAL
read (P^.apellido);
p^.edad := 30;
p^.altura := 1.74;
peso^ := 3;  read (peso^) BIEN
frase^ := "La casa de María"
read (frase^) BIEN
```



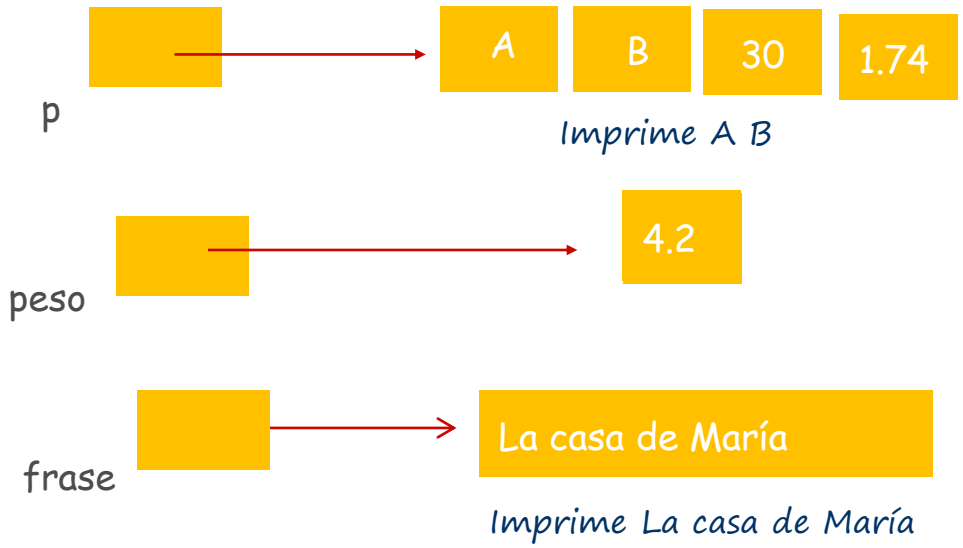
TIPO DE DATO PUNTERO - OPERACIONES

- Acceso a la información de la variable referenciada

```
Type
 TipoString = string[20];
 Datos = record
          Nombre: TipoString;
          Apellido:TipoString;
          Edad: integer;
          Altura: real
         end;
 PtrDatos = ^datos;
 PtrReal = ^real;
 PtrString = ^TipoString;

Var
   p, q : PtrDatos;
   peso : PtrReal;
   frase : PtrString;
```

```
Begin
……..
write (P^.nombre);
write (P^.apellido);
peso^:=peso^ + 1.2;
write ( frase^);
```

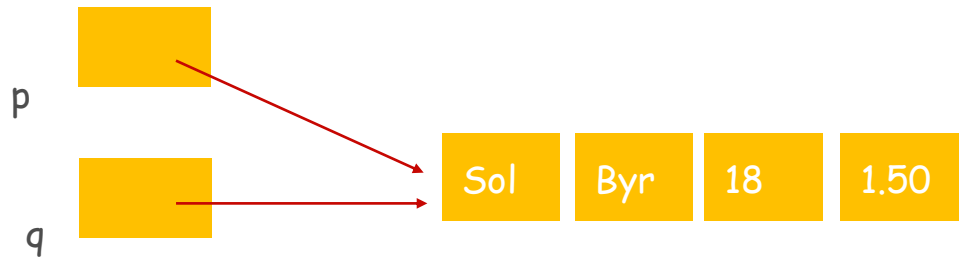


TIPO DE DATO PUNTERO - OPERACIONES

- Asignación de punteros

```
Type
 TipoString = string[20];
 Datos = record
          Nombre: TipoString;
          Apellido: TipoString;
          Edad: integer;
          Altura: real
         end;
 PtrDatos = ^datos;

Var
  p, q : PtrDatos;
```



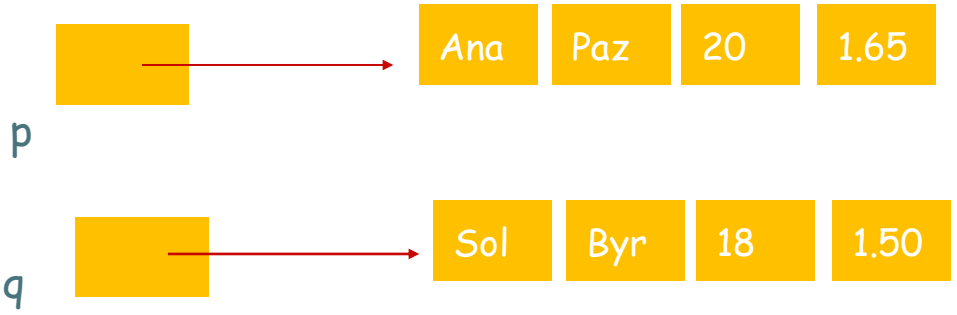

```
New (q);
Read(q^.nombre, q^.apellido, q^.edad, q^.altura);

p := q
```

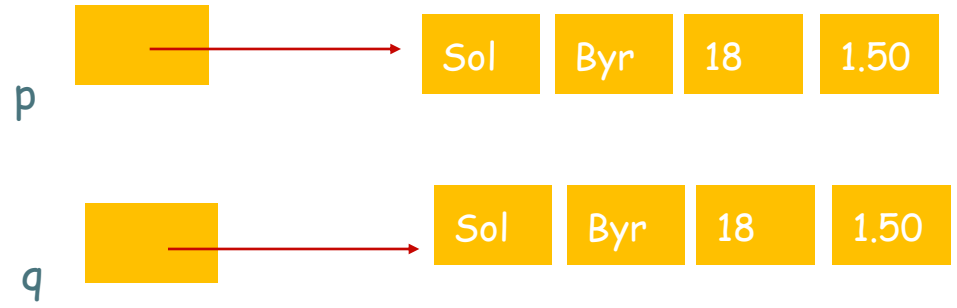
TIPO DE DATO PUNTERO - OPERACIONES

- Asignación a la variable referenciada (asignación de objetos “referenciados”)

Si se tiene p y q como se muestra



y se hace `p^ := q^`



(punteros distintos apuntando a valores iguales)

TIPO DE DATO PUNTERO - OPERACIONES

- Operaciones de comparación

Pueden aparecer en expresiones relacionales como: p = q y p <> q

- Operaciones de Entrada / Salida

No se puede leer y/o escribir una variable puntero. Si se puede leer y/o escribir los objetos que ellos referencian dependiendo del tipo apuntado como ya se vió.

TIPO DE DATO PUNTERO - OPERACIONES

- Eliminación de la variable referenciada → **Dispose (p)**

Si se tiene

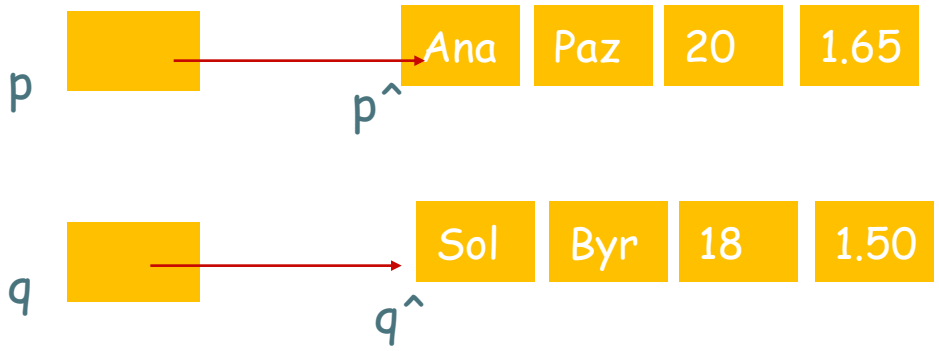
Y se hace dispose (p); el efecto es que se "rompe" el enlace entre p y p^. No es posible volver a trabajar con el dato direccionado por p, por lo tanto, ese espacio de memoria puede ser "reutilizado".

El contenido del puntero p queda indeterminado. No se lo puede utilizar a menos que se lo asigne nuevamente.

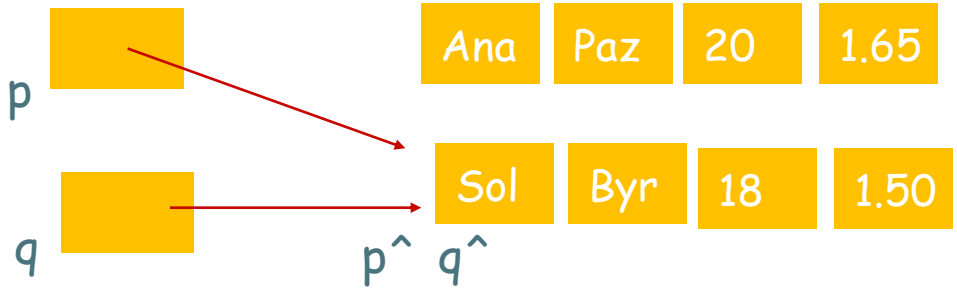
TIPO DE DATO PUNTERO – EFECTO DE LA OPERACIÓN DISPOSE

¿Qué ocurre cuando se usa el procedimiento Dispose y cuando no se lo usa?

Si se tiene:



Y se hace `p := q;` el efecto es:



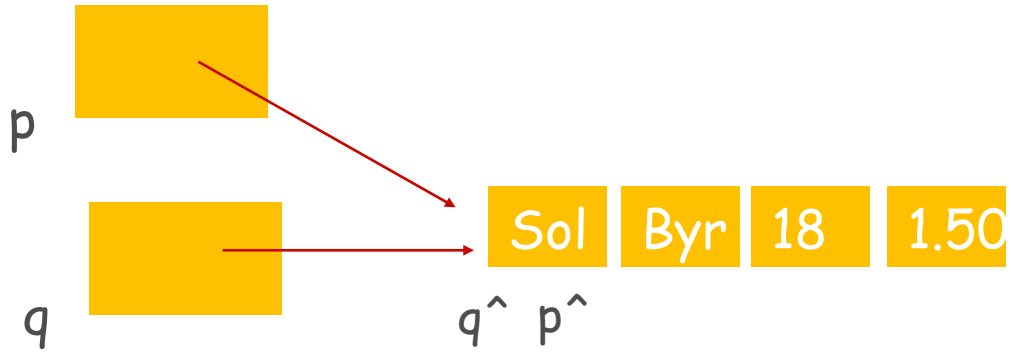
➡ El espacio de memoria referenciado por p sigue "ocupado", pero no es posible referenciarlo.

TIPO DE DATO PUNTERO – EFECTO DE LA OPERACIÓN DISPOSE

Si en cambio se hace:

```
Dispose (p);
p := q;
```

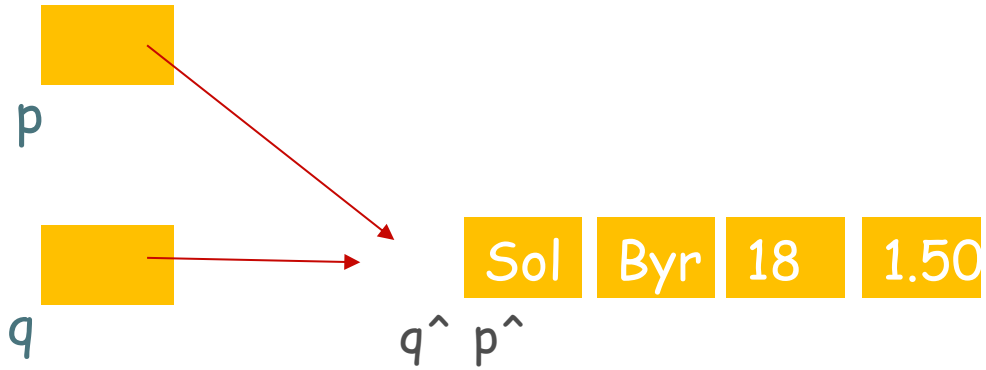
El efecto es:



- Como el espacio de memoria referenciado por p fue “liberado”, entonces puede ser reutilizado.

TIPO DE DATO PUNTERO – EFECTO DE LA OPERACIÓN DISPOSE

Supongamos que:



¿Qué ocurre si se hace

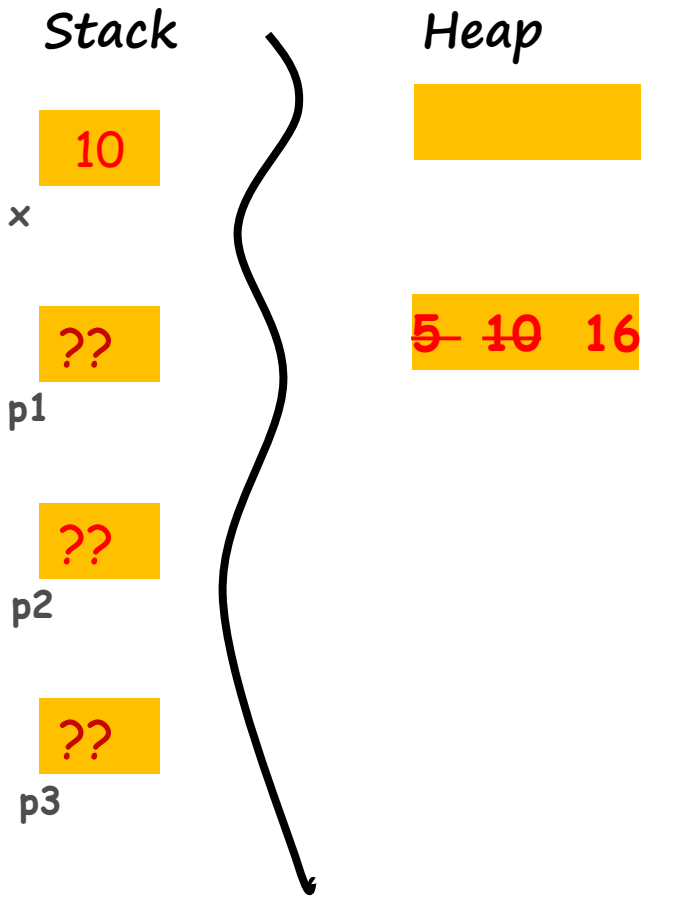
Dispose (p)?

- El espacio de memoria referenciado por ese puntero será “liberado”, por lo tanto, NINGÚN otro puntero que esté referenciando esa dirección podrá utilizarla.

TIPO DE DATO PUNTERO – EJERCITACION

```
Type
  pint= ^integer;

var  x : integer;
       p1, p2, p3: pint;
begin
 read (x) ;
 new (p1) ;
 new(p2) ;
 p1^ := x ;
 p2^ := p1^ + 1 ;
 read (x) ;
 p1^:= x ;
 p3 := p1 ;
 p1^ := p1^+p2^;
 writeln ('Elemento en p1: ', p1^);16
 writeln ('Elemento en p2: ', p2^);6
 writeln ('Elemento en p3: ', p3^);16
 NEW (P1); new (p3);p1:= nil; p1:= p3;
End.
```



¿Qué pasa si...?

p1:=Nil?

Dispose (p2)?

Dispose (P1)?

TIPO DE DATO PUNTERO – EJERCITACION ¿Qué imprime?

```
program punterosC;
type
  cadena = string[50];
  puntero_cadena = ^cadena;

procedure cambiar(var pun1: puntero_cadena;
                  pun2: puntero_cadena);
begin
  pun1:= pun2;
end;

var
  p1, p2: puntero_cadena;
begin
  new(p1);
  p1^:= 'Hoy es lunes';
  writeln('El contenido de p1^: ', p1^);
  cambiar(p2, p1);
  writeln('El contenido de p1^: ', p1^);
  writeln('El contenido de p2^: ', p2^);
end.
```