



Conceptos de Algoritmos Datos y Programas

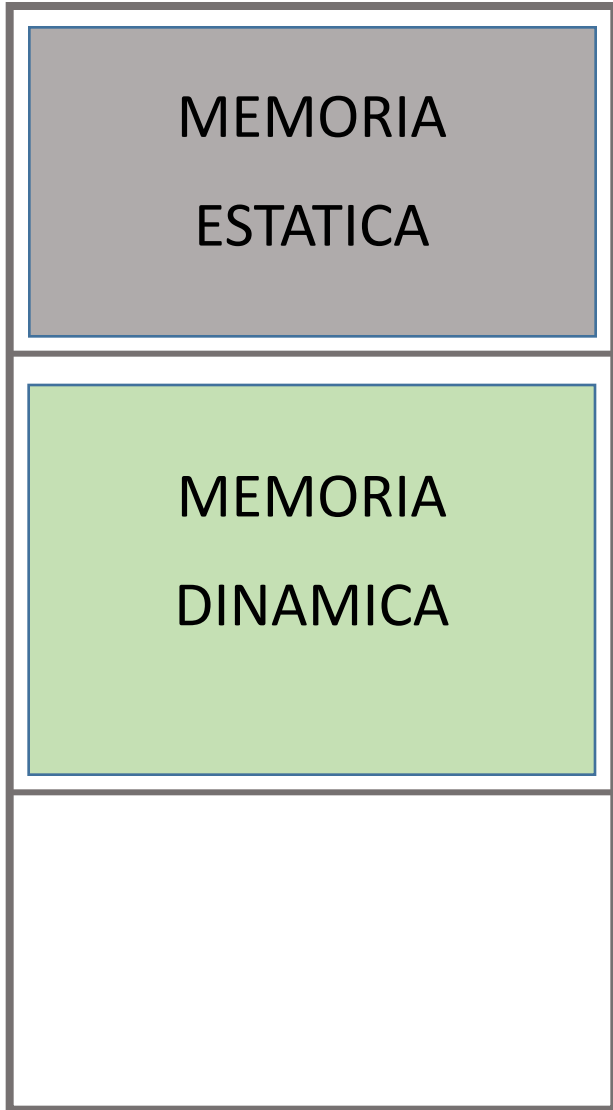


CADP – TEMAS



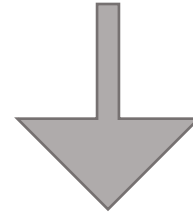
- Alocación estática – Alocación dinámica
- Tipo de datos PUNTERO

CADP – TIPO DE DATO PUNTERO

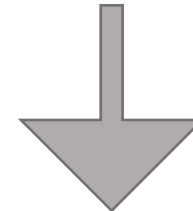


char, boolean,
integer, real,
string, subrango,
registro, vector
PUNTERO

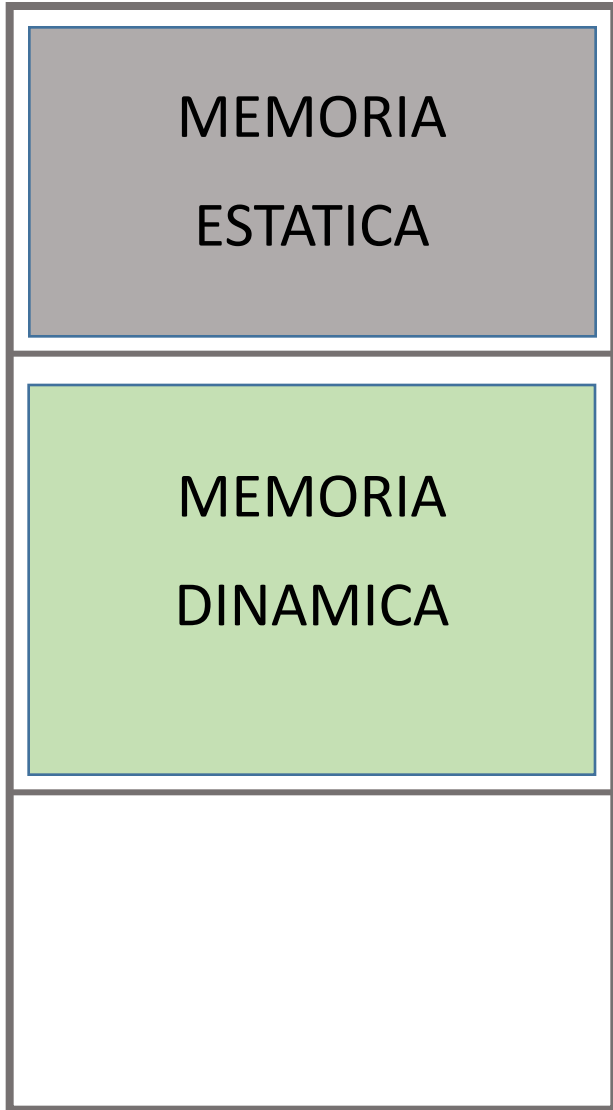
Hasta ahora, cualquier variable que se declare en un programa es alojada en la memoria estática de la CPU



Una variable de tipo puntero contiene como dato una dirección de memoria dinámica.



En esa dirección de memoria se encuentra el dato que realmente se quiere guardar.



Tipo de variable	Bytes que ocupa
Char	1 byte
Boolean	1 byte
Integer	6 bytes
Real	8 bytes
String	Tamaño + 1 (sino se especifica el tamaño es 255 + 1)
Subrango	Depende el tipo
Registro	La suma de sus campos
Vector	Dimensión física * tipo elemento
Puntero	4 bytes

CADP – TIPO DE DATO PUNTERO



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.

TIPO DE DATO

SIMPLE

COMPUESTO

DEFINIDO POR EL LENGUAJE

DEFINIDO POR EL PROGRAMADOR

DEFINIDO POR EL LENGUAJE

DEFINIDO POR EL PROGRAMADOR

Integer
Real
Char
Boolean
Puntero

Subrango

String

Registros

Arreglos

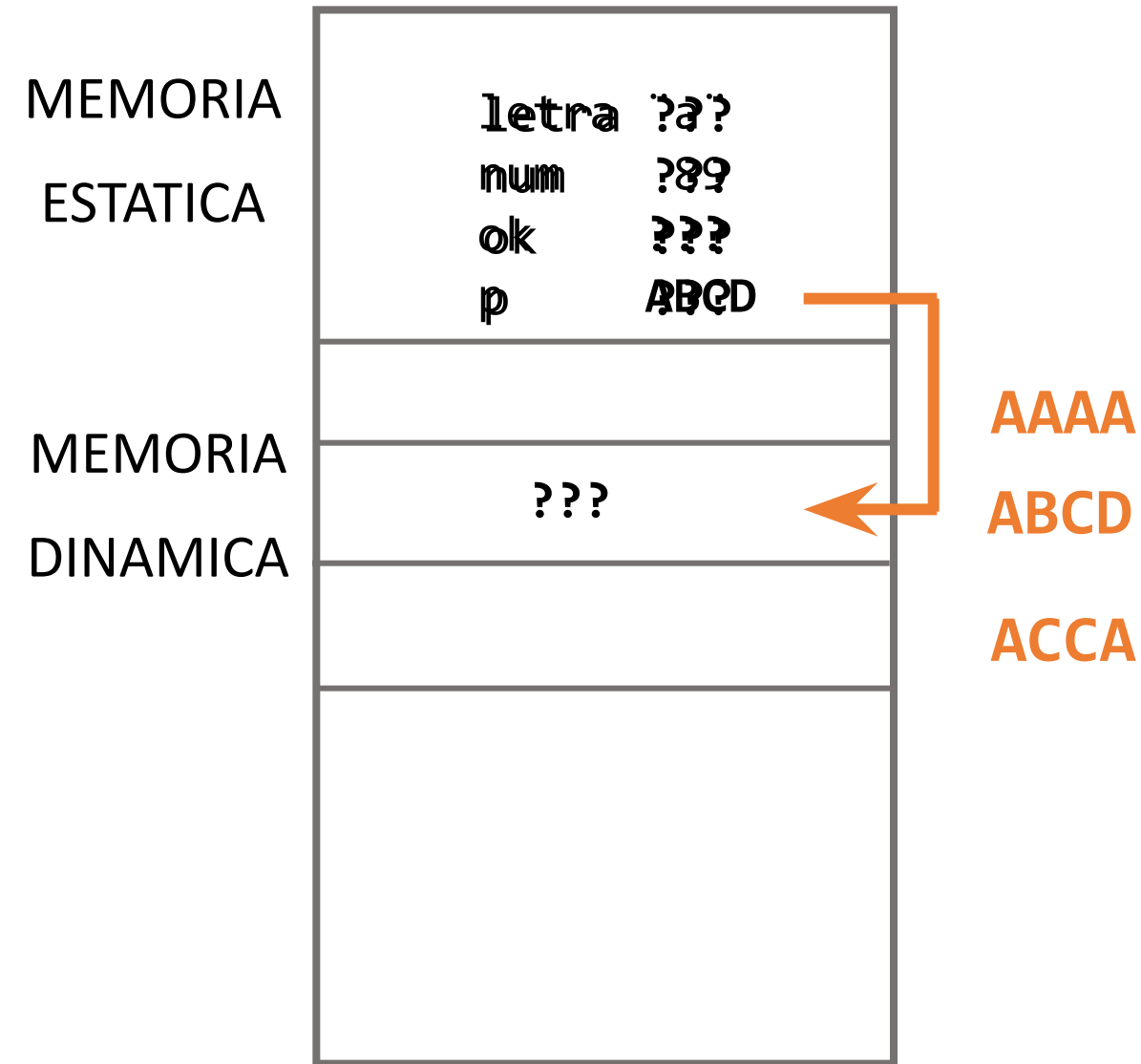
CADP – TIPO DE DATO **PUNTERO**



Es un tipo de variable usada para almacenar una dirección en memoria dinámica. En esa dirección de memoria se encuentra el valor real que almacena. El valor puede ser de cualquiera de los tipos vistos (char, boolean, integer, real, string, registro, arreglo u otro puntero).

Un puntero es un tipo de datos simple.

**Cómo se ve
gráficamente?**



Type

puntero //ya veremos como

Var

```
letra:char;
num:integer;
ok:boolean;
p:puntero;
```

**Cómo se
declaran?**

Begin

```
letra:= "a";
num:= 89;
p: pide memoria //ya veremos como
```

...

End.

CADP – TIPO DE DATO PUNTERO

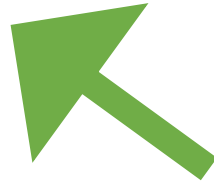


Type

puntero = ^ tipo de datos;

Var

p:puntero;



Puede ser cualquiera de los tipos vistos previamente: integer, boolean, char, real, subrango, registro, vector.

Ejemplos

CADP – TIPO DE DATO PUNTERO



Type

```
TipoCadena = array [1..10] of char;
```

```
PunCadena = ^TipoCadena;
```

```
PunReal = ^real;
```

```
PunString = ^string;
```

```
Datos = record  
    nombre: string[10];  
    apellido: string[10];  
    altura: real;  
end;
```

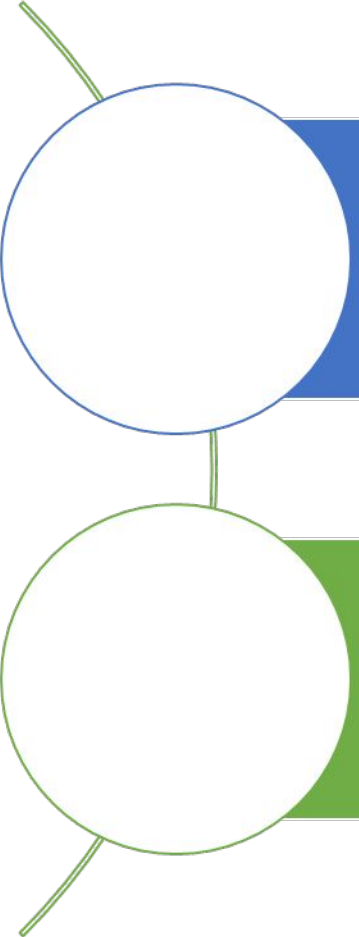
```
PunDatos = ^datos;
```

```
var  
    pReal: PunReal;  
    t: PunString;  
    r: PunString;  
    puntero: PunCadena;  
    p,q: PunDatos;  
    d:datos;
```

```
begin  
    ....  
end.
```



Cómo vamos a trabajar?



Una variable de tipo puntero ocupa una cantidad de memoria fija, independiente del tipo de dato al que apunta (4 bytes). Es un tipo de datos simple.

Una variable de tipo puntero puede reservar y liberar memoria durante la ejecución de un programa para almacenar su contenido

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.

CADP – TIPO DE DATO **PUNTERO**



Creación de una variable puntero.

Destrucción de una variable puntero.

Asignación entre variables puntero.

Asignación de un valor al contenido de una variable puntero.

Comparación de una variable puntero



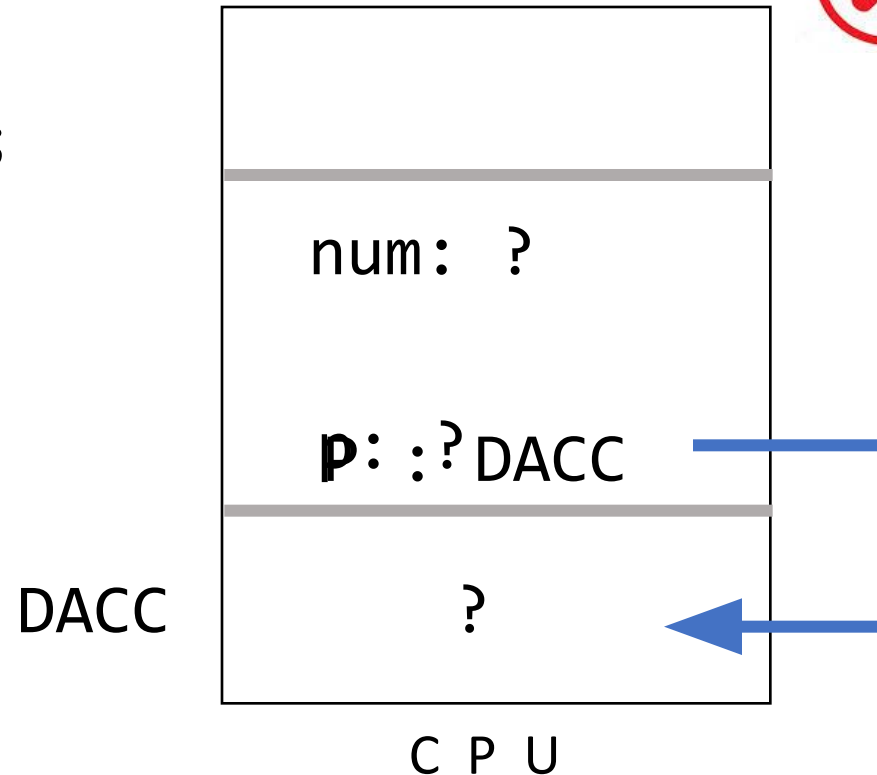
CADP – TIPO DE DATO PUNTERO



CREACION

Implica reservar una dirección memoria dinámica libre para poder asignarle contenidos a la dirección que contiene la variable de tipo puntero. **new(variable tipo puntero)**

```
Program uno;  
Type  
  puntero = ^integer;  
Var  
  num:integer;  
  p:puntero;  
  
Begin  
  new (p);  
  ...  
End.
```



No se puede asignar a
un puntero una
dirección específica
(p:= ABCD)



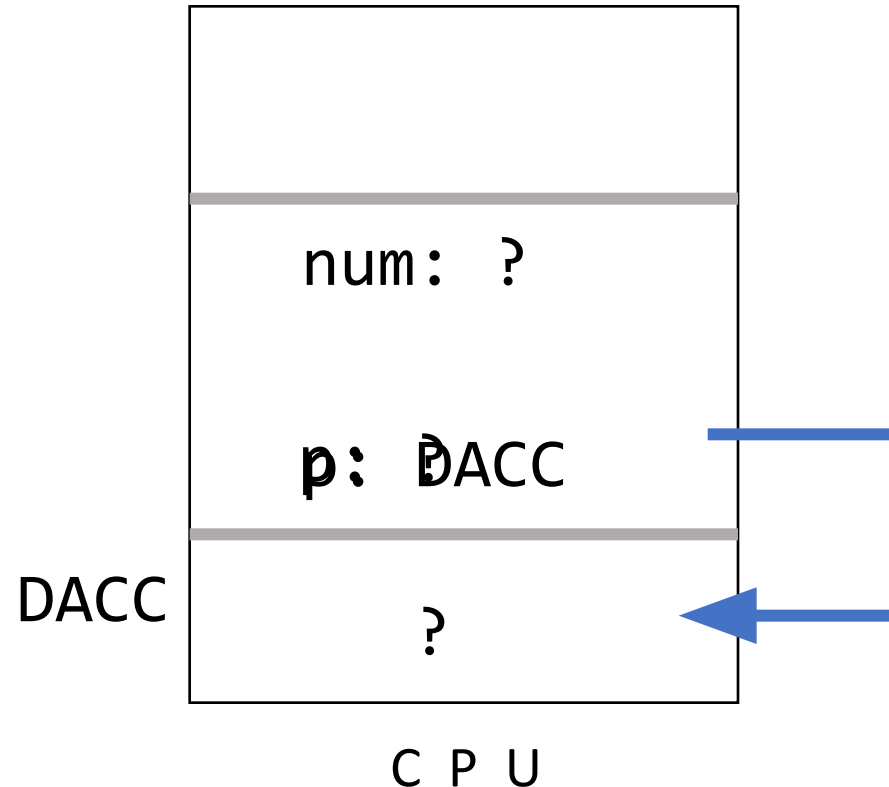
ELIMINACION

Implica liberar la memoria dinámica que contenía la variable de tipo puntero. **dispose(variable tipo puntero)**

```
Program uno;  
Type  
    puntero = ^integer;
```

```
Var  
    num:integer;  
    p:puntero;
```

```
Begin  
    new (p);  
    dispose (p);  
End.
```



CADP – TIPO DE DATO PUNTERO



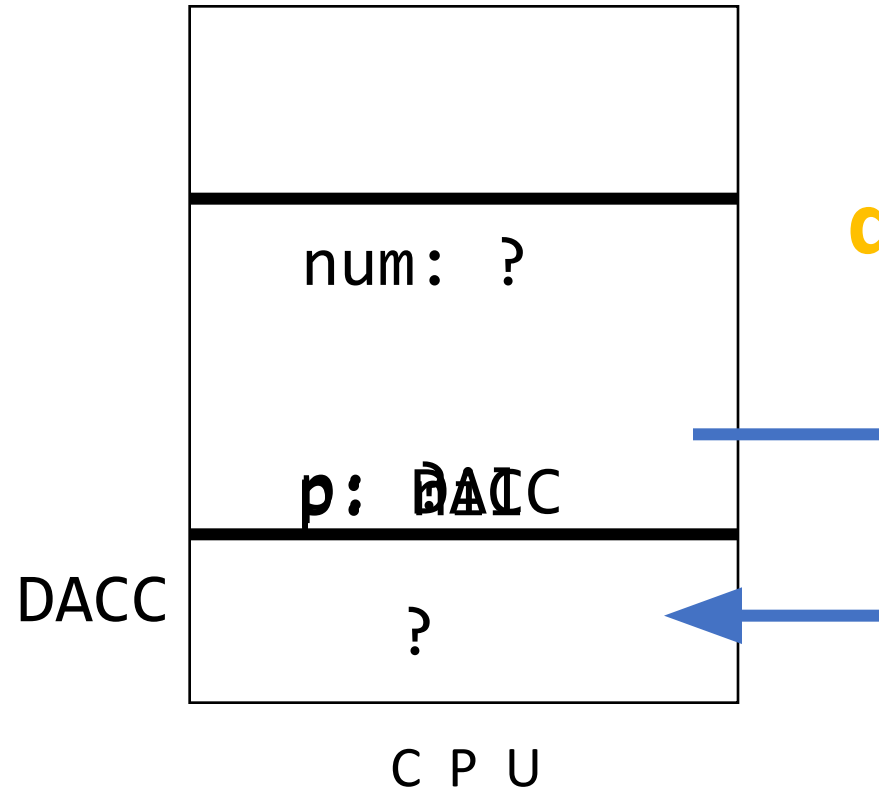
LIBERACION

Implica cortar el enlace que existe con la memoria dinámica. La misma queda ocupada pero ya no se puede acceder. **nil**

```
Program uno;  
Type  
  puntero = ^integer;
```

```
Var  
  num:integer;  
  p:puntero;
```

```
Begin  
  new (p);  
  p:= nil;  
End.
```



Cuál es la diferencia?



DISPOSE (p)

Libera la conexión que existe entre la variable y la posición de memoria.

Libera la posición de memoria.

La memoria liberada puede utilizarse en otro momento del programa.



$p := \text{nil}$

Libera la conexión que existe entre la variable y la posición de memoria.

La memoria sigue ocupada.

La memoria no se puede referenciar ni utilizar.

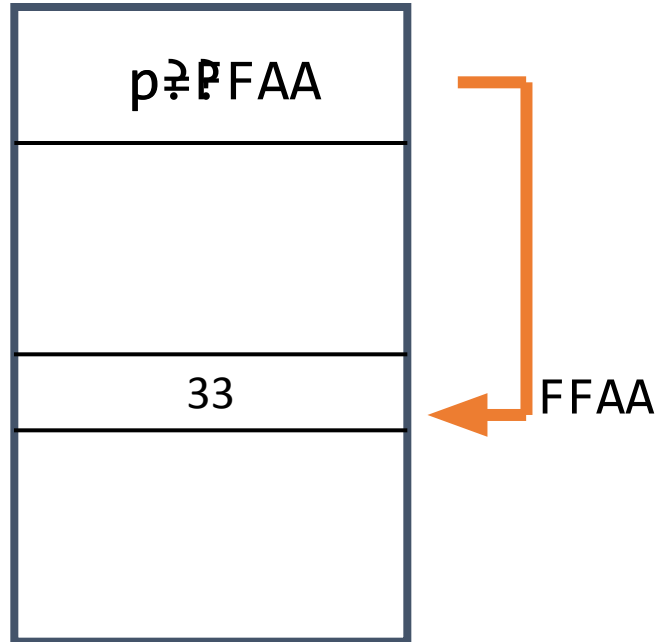
Gráficamente ...?



DISPOSE (p)

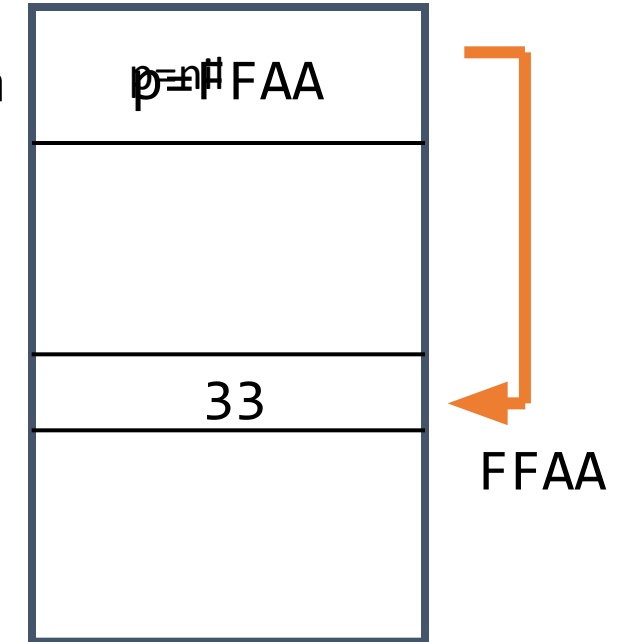
`p := nil`

Memoria



DISPOSE (P)

Memoria



`P := nil`

CADP – TIPO DE DATO PUNTERO



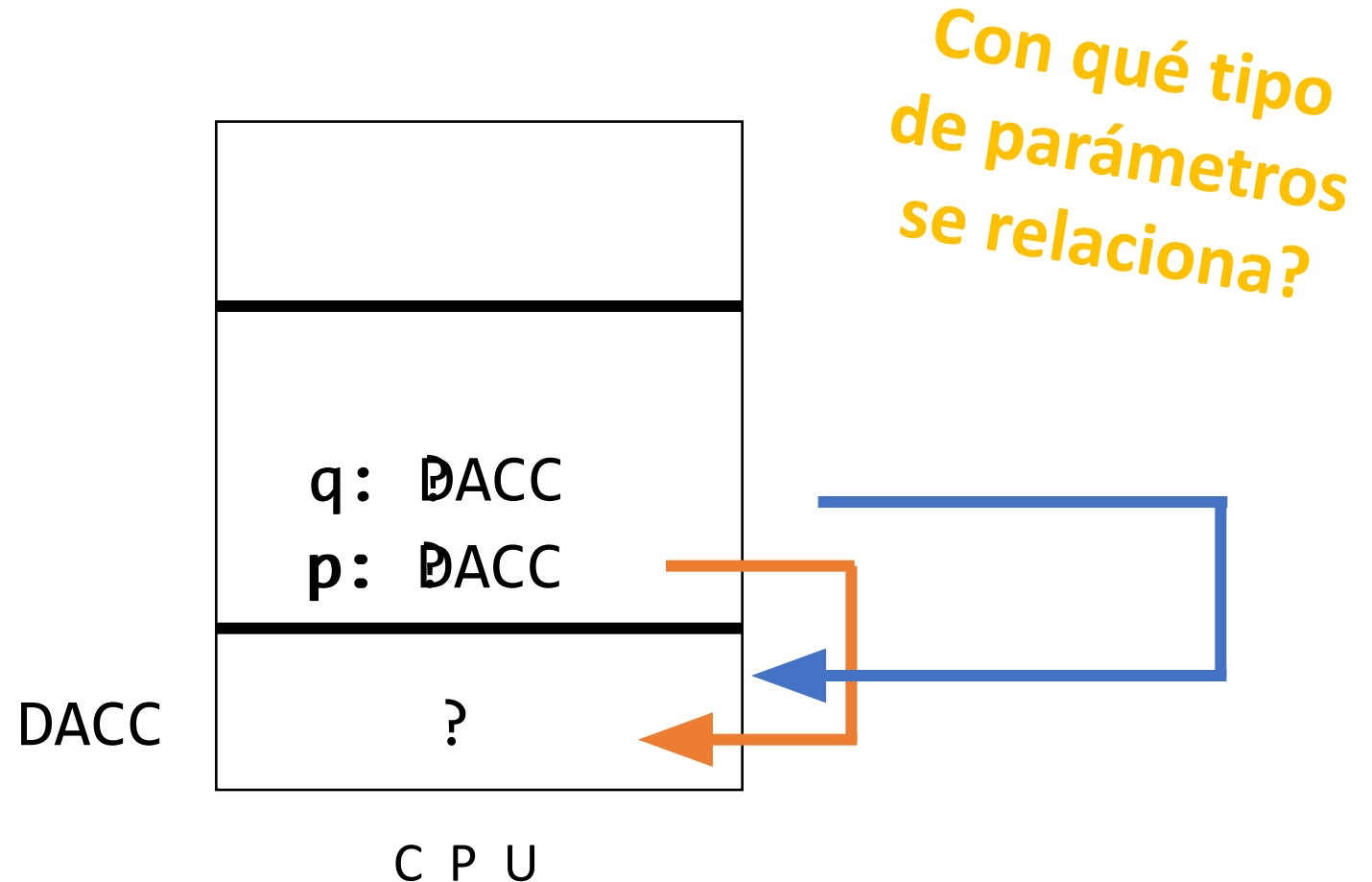
ASIGNACION entre punteros

Implica asignar la dirección de un puntero a otra variable puntero del mismo tipo. **:=**

```
Program uno;  
Type  
  puntero = ^integer;
```

```
Var  
  q:puntero;  
  p:puntero;
```

```
Begin  
  new (p);  
  q:=p;  
End.
```





```
Program uno;
```

```
Type
```

```
    puntero = ^integer;
```

```
Var
```

```
    q:puntero;
```

```
    p:puntero;
```

```
Begin
```

```
    new (p);
```

```
    q:=p;
```

```
    dispose (p);
```

```
End.
```

**Cómo queda la
memoria en
cada programa?**

```
Program dos;
```

```
Type
```

```
    puntero = ^integer;
```

```
Var
```

```
    q:puntero;
```

```
    p:puntero;
```

```
Begin
```

```
    new (p);
```

```
    q:=p;
```

```
    p:= nil;
```

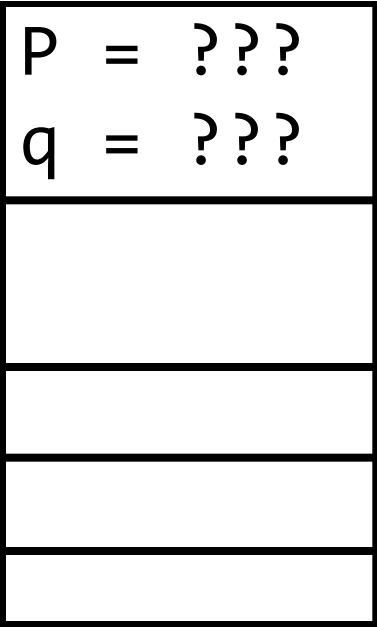
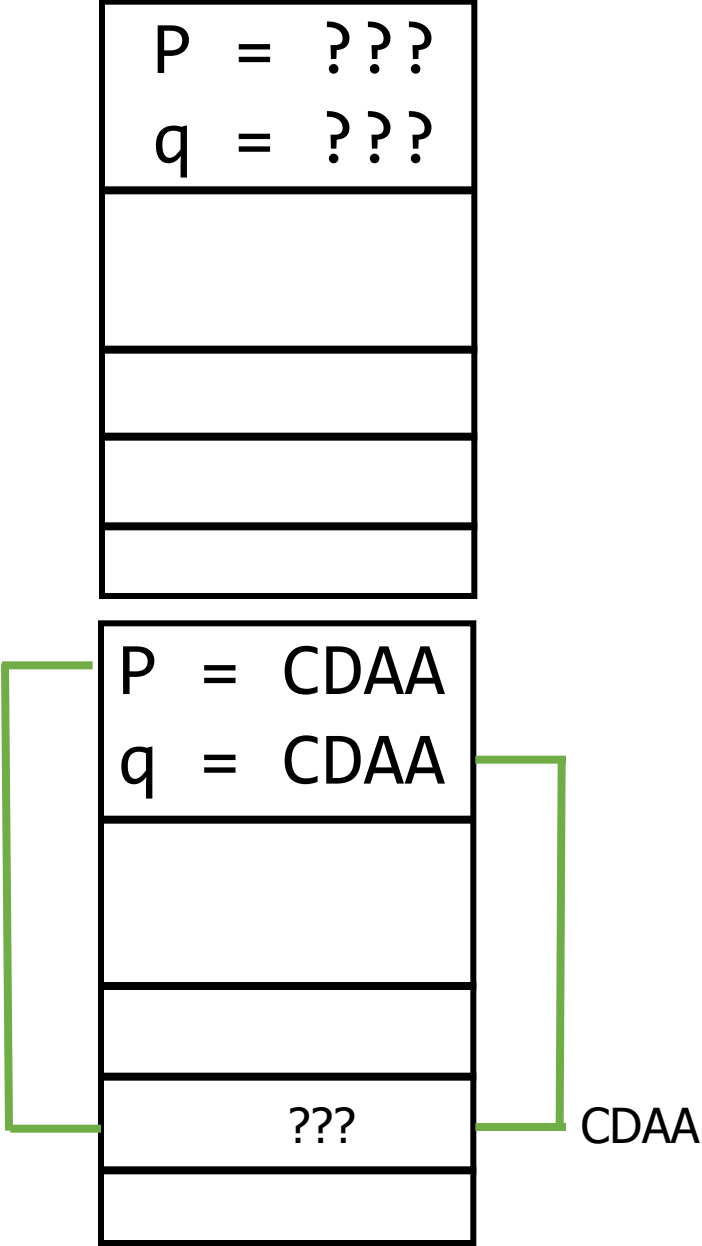
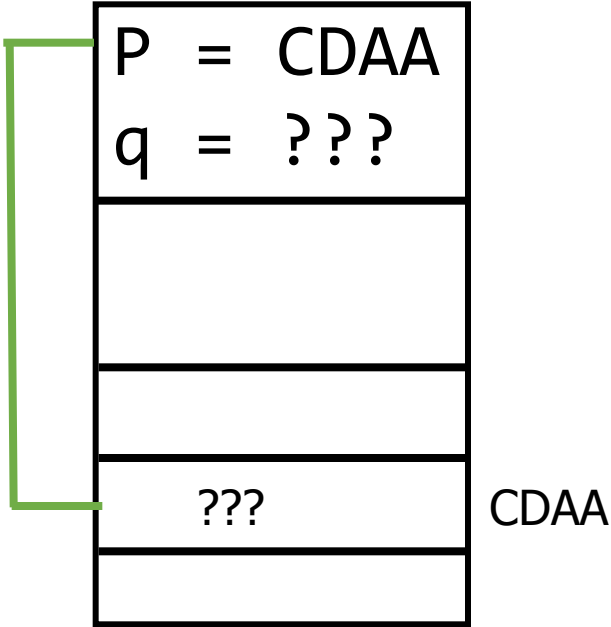
```
End.
```

CADP – TIPO DE DATO

PUNTERO



```
Var
  p,q:pun;
Begin
  new (p);
  q:=p;
  dispose(p);
End.
```

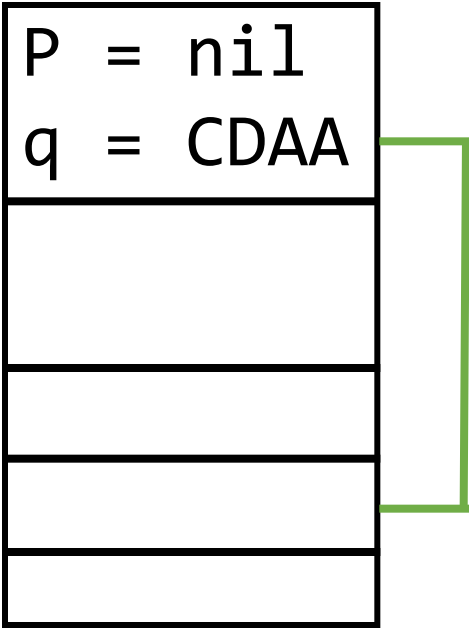
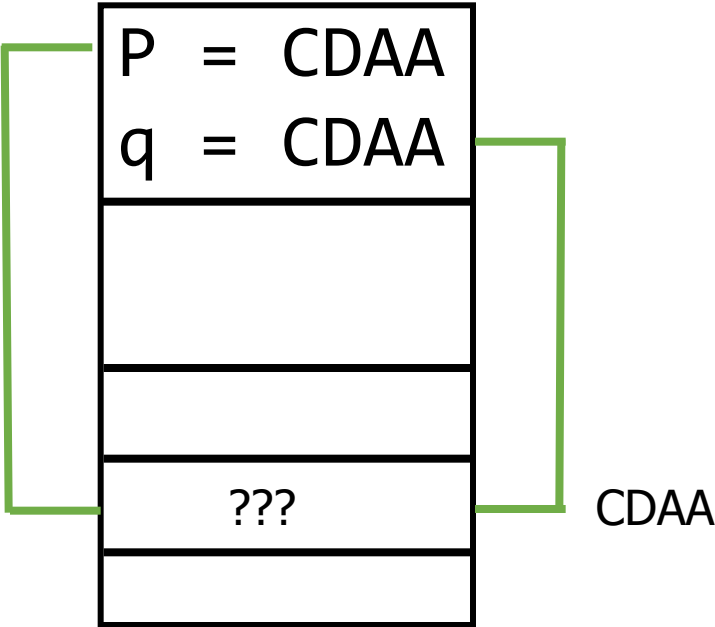
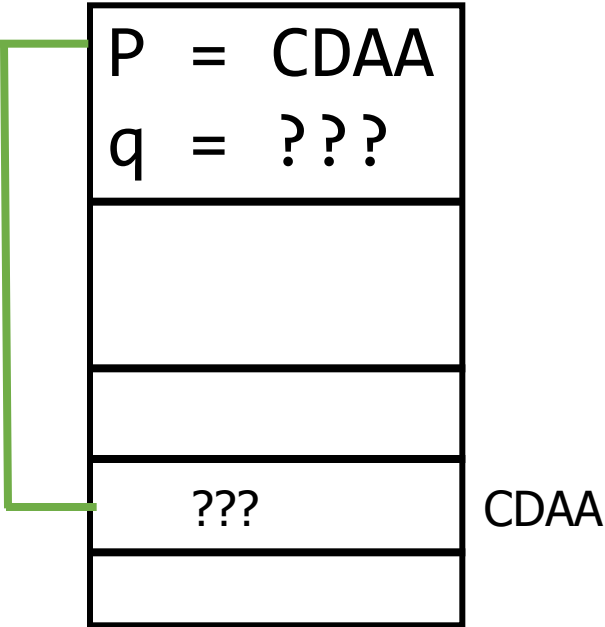
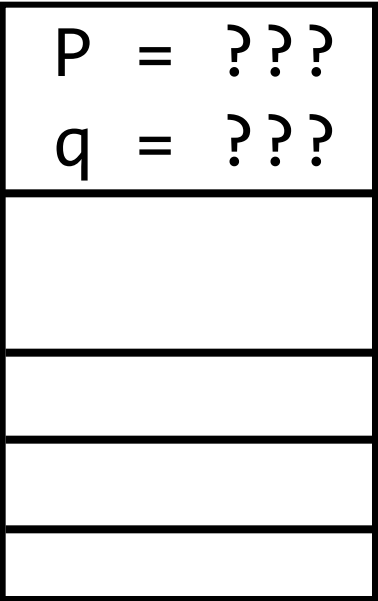


CADP – TIPO DE DATO

PUNTERO



```
Var
  p,q:pun;
Begin
  new (p);
  q:=p;
  p:= nil;
End.
```



CADP – TIPO DE DATO PUNTERO



CONTENIDO de un puntero

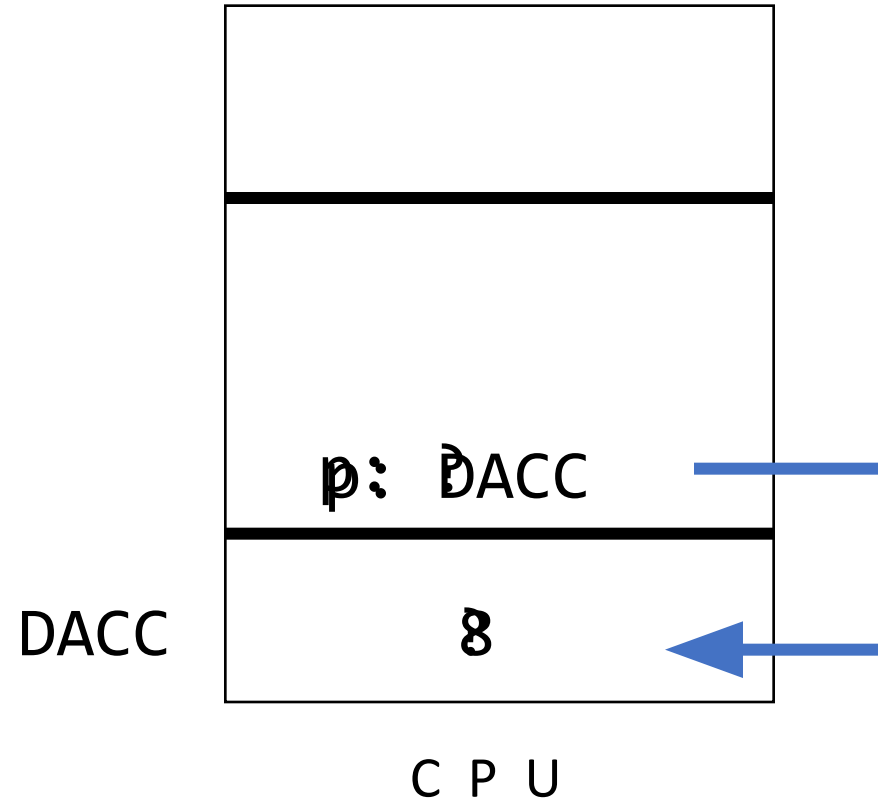
Implica poder acceder al contenido que contiene la dirección de memoria que tiene una variable de tipo puntero. ^

```
Program uno;  
Type  
  puntero = ^integer;
```

```
Var  
  p:puntero;
```

```
Begin  
  new (p);  
  p^:=8;
```

```
End.
```



Qué operaciones
podré hacer?

Ejemplos ...

CADP – TIPO DE DATO

PUNTERO



EJEMPLOS – Cómo varía la memoria?

Qué imprime cada programa?

```
Program uno;  
Type  
  punt = ^integer;  
Var  
  p,q:punt;  
  num:integer;  
  
Begin  
  num:= 63;  
  new (p);  
  new(q);  
  q^:= num - 10;  
  
  write(q^);  
  write(p^);  
end.
```

```
Program dos;  
Type  
  punt = ^integer;  
Var  
  p,q:punt;  
  
Begin  
  new (p);  
  p^:= 14;  
  write (p^);  
  q:=p;  
  q^:= q^*10;  
  write (p^);  
  write(q^);  
  dispose (q);  
  write (p^);  
  write (q^);  
end.
```

```
Program tres;  
Type  
  punt= ^integer;  
Var  
  p,q:punt;  
  
Begin  
  new (p);  
  new(q);  
  p:= q;  
  q^:=10;  
  
  write(q^);  
  write(p^);  
end.
```

```
Program cuatro;  
Type  
  punt = ^integer;  
Var  
  p,q:punt;  
  
Begin  
  new (p);  
  p^:= 14;  
  write (p^);  
  q:=p;  
  q^:= q^*10;  
  write (p^);  
  write(q^);  
  q=nil;  
  write (p^);  
  write(q^);  
End.
```



- if ($p = \text{nil}$) then, compara si el puntero p no tiene dirección asignada.
- if ($p = q$) then, compara si los punteros p y q apuntan a la misma dirección de memoria.
- if ($p^\wedge = q^\wedge$) then, compara si los punteros p y q tienen el mismo contenido.
- no se puede hacer $\text{read}(p)$, ni $\text{write}(p)$, siendo p una variable puntero.
- no se puede asignar una dirección de manera directa a un puntero, $p := \text{ABCD}$
- no se pueden comparar las direcciones de dos punteros ($p < q$).