PROGRAMACIÓN I

TEORÍA – CECILIA SANZ

Temas

- ✓ Alocación estática y dinámica
- ✓ Concepto de Punteros
- ✓ Ejemplos

Motivación


Mecanismos para la reserva de la memoria
La metáfora del teatro…

Alocación Estática

Planteemos una metáfora en la que tengo una serie de sillas disponibles que representan celdas de memoria.

En el caso de un vector, cuando declaramos una variable supongamos:

Type vector=array [1..3] of persona;
Var v: vector;

Se reserva espacio para v y esto se mantiene mientras v “tenga vida” (recordar concepto de tiempo de vida)

Alocación Dinámica

Guardo un dato en el espacio reservado

Supongamos ahora que no se admiten reservas, y se usan las sillas a medida que llegan las personas y las libera cuando ya no las necesitan.

Esta modalidad de reserva de la memoria se conoce como dinámica.

Alocación estática versus dinámica

ALOCACION ESTÁTICA Y DINÁMICA

Hasta ahora trabajábamos con alocación estática:

- las variables reservan memoria en su declaración
- permite al lenguaje hacer validaciones de tipo en tiempo de compilación.

PROBLEMAS DE LA ALOCACION ESTATICA

La rigidez ya que no permite que las estructuras varíen su dimensión.

ALOCACION ESTÁTICA Y DINÁMICA

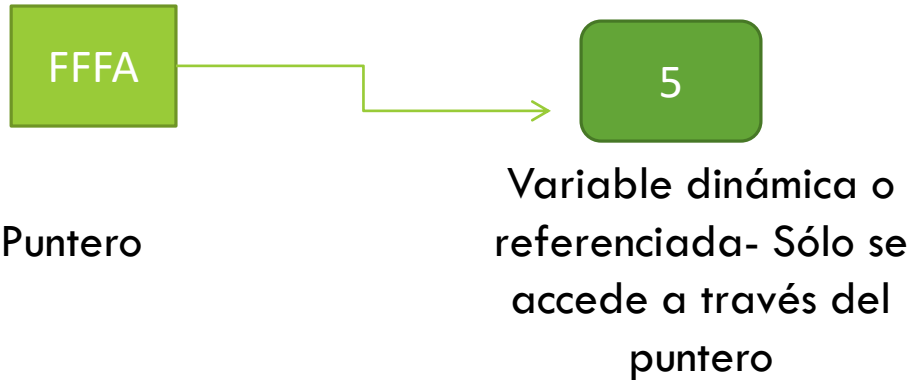
Alocación dinámica: permite declarar variables dinámicas o referenciadas. En Pascal existe un tipo de datos **PUNTERO** que permite generar referencias a variables dinámicas.

Un puntero es un tipo de variable usada para almacenar la dirección en memoria de otra variable, en lugar de un valor convencional.

Ejemplo de punteros

ALOCACION ESTÁTICA Y DINÁMICA

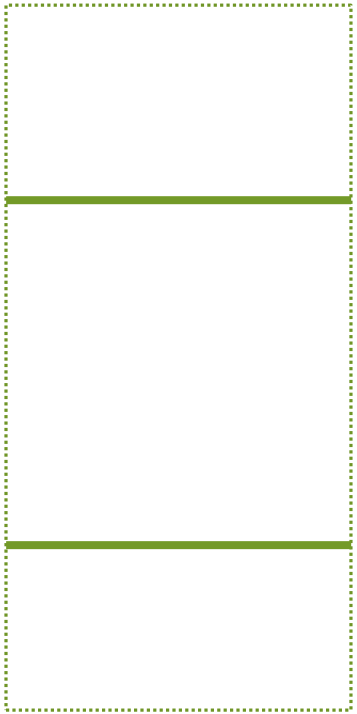
Mediante la variable de tipo puntero se accede a esa otra variable, almacenada en la dirección de memoria que señala el puntero. Es decir, el valor de la variable de tipo puntero es una dirección de memoria.



ALOCACION ESTÁTICA Y DINÁMICA

- Se dice que el puntero apunta o señala a la variable almacenada en la dirección de memoria que contiene el puntero.
- Lo que interesa es el dato contenido en esa variable apuntada.
- No hay que confundir la variable apuntada con el puntero.

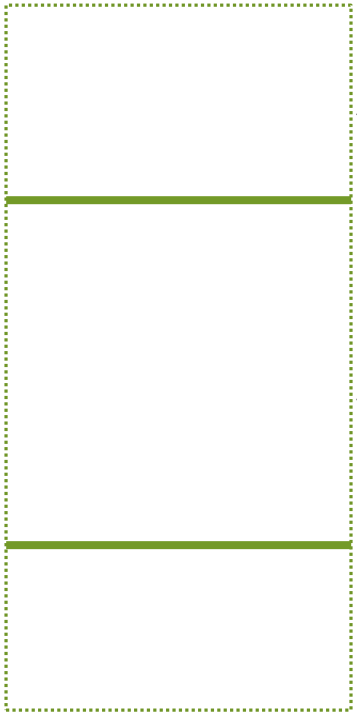
ALOCACION ESTÁTICA Y DINÁMICA



Memoria dinámica

Memoria estática
- variables integer, char, real, boolean,
- registro, arreglos.
- Tamaño de la memoria fijo

ALOCACION ESTÁTICA Y DINÁMICA



Memoria dinámica

Memoria estática

64K= 65536 bytes

Tamaño de las variables:

Char = 1byte

Integer = 2 bytes

Real = 6 bytes

Boolean = 1 byte

String = cantidad de caracteres + 1

Registro = la suma de lo que ocupa c/ campo

Puntero = 4 bytes

¿Qué ocurre si declaro un arreglo de 3000 strings?

768000 bytes

Punteros

ALOCACION ESTÁTICA Y DINÁMICA

Un tipo puntero se define, en PASCAL, de la siguiente forma:

```
Program uno;
    TYPE TipoPuntero= ^TipoVariableApuntada;
```

```
Program uno;
    TYPE
      PunteroAEntero  =  ^Integer;
    VAR
      Puntero:  PunteroAEntero;
```

Un puntero puede apuntar a cualquier tipo.

Punteros

- Un puntero es un **tipo de dato simple** que contiene la dirección de otro dato.
- Los punteros pueden apuntar **solamente a variables dinámicas**, es decir, a datos que están almacenados en memoria dinámica (heap).
- Cada variable de tipo puntero puede apuntar a un único tipo de dato.
- **Una variable de tipo puntero ocupa 4 byte de memoria (stack) para su representación interna en Pascal.**

PUNTEROS

```
Type
PtrReal = ^Real;
Cadena= string[20];
StringPtr = ^Cadena;
Vector = array [1..10] of char;
PtrVector = ^Vector;
Datos = record
            Nombre: string[10];
            Apellido: string[10];
            Altura: real;
           End;
PtrDatos = ^datos;
```

```
Declaración de variables
  var
     PunteroReal : PtrReal;     (o ^real)
     t : StringPtr;
     r : StringPtr;
     s : Cadena;
     Puntero : PtrVector;
     p, q : PtrDatos;
```

Importante: Pascal permite que las funciones retornen punteros, por ejemplo, FunctionVerificar (s:Cadena): StringPtr;

PUNTEROS

- Una variable de tipo puntero ocupa una cantidad de memoria fija, 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 emplear un tipo de dato que permita referenciar nuevas posiciones de memoria que no han sido reservadas a priori, y que se van a crear y destruir en tiempo de ejecución.

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

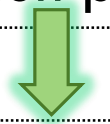
PUNTEROS

```
Program dos;

Type
  puntero = ^integer;


Var
  p: puntero;
```

¿Cómo trabajo con punteros?



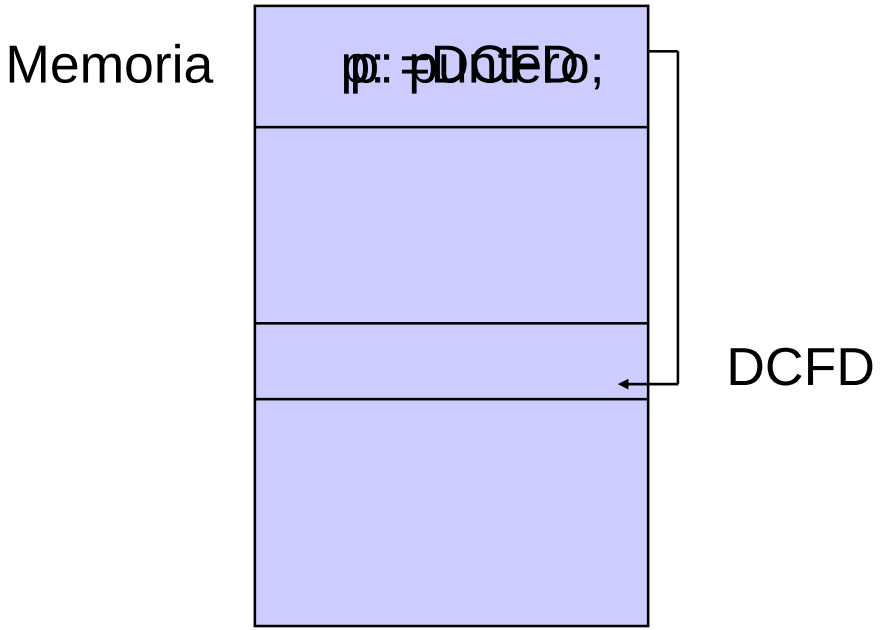
- Creación de una variable referenciada.
- Destrucción de una variable referenciada.
- Asignación entre variables puntero.
- Asignación de un valor al contenido de una variable puntero.
- Comparación de una variable puntero

Operación NEW

PUNTEROS – Creación de variable apuntada

Para crear una variable de tipo dinámico hay que reservar memoria. Para ello Pascal nos provee la operación NEW

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

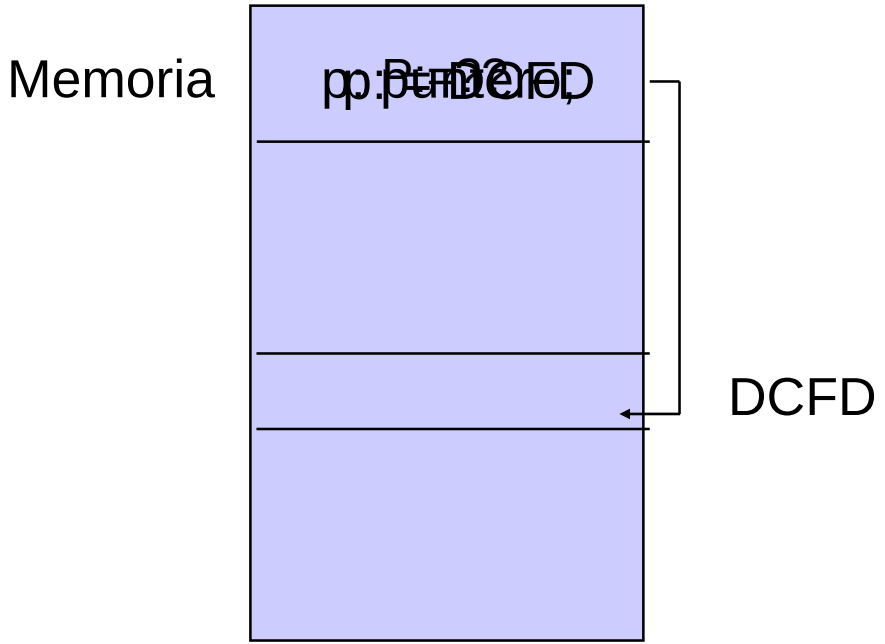


Operación DISPOSE

PUNTEROS – Liberar la variable referenciada

La variable dinámica o referenciada una vez que no se utiliza más debe liberarse la memoria

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



Operación ASIGNACIÓN DE PUNTEROS

PUNTEROS - ASIGNACION

A una variable de tipo puntero se le permite asignar dos valores:

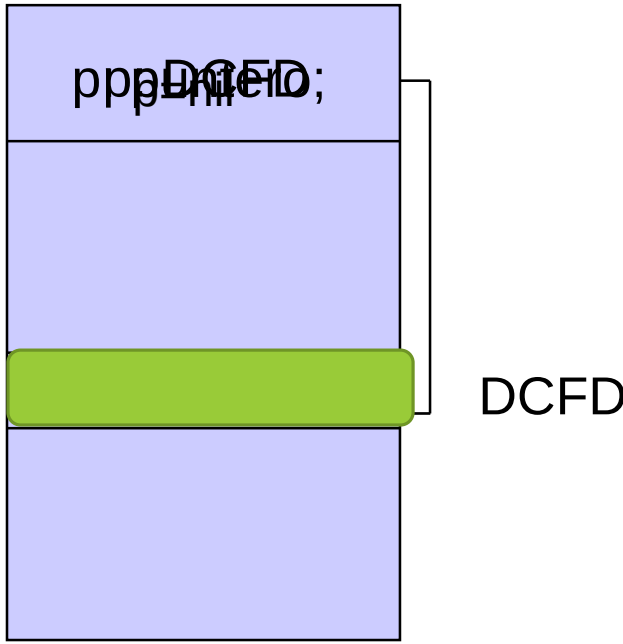
- el valor nil.
- otra variable de tipo puntero.

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

Al no hacer dispose, la zona de memoria permanece reservada

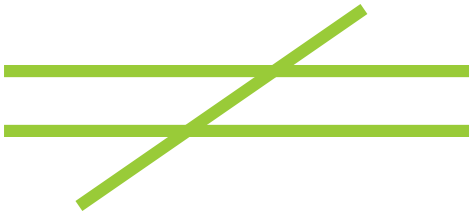
¿Qué diferencia existe con el dispose?

Memoria



PUNTEROS - ASIGNACION

El dispose elimina “la conexión” que existe entre la variable puntero y su dirección y luego libera esa dirección; por lo tanto esa dirección podría ser reasignada luego (por medio del new a otro puntero o al mismo).

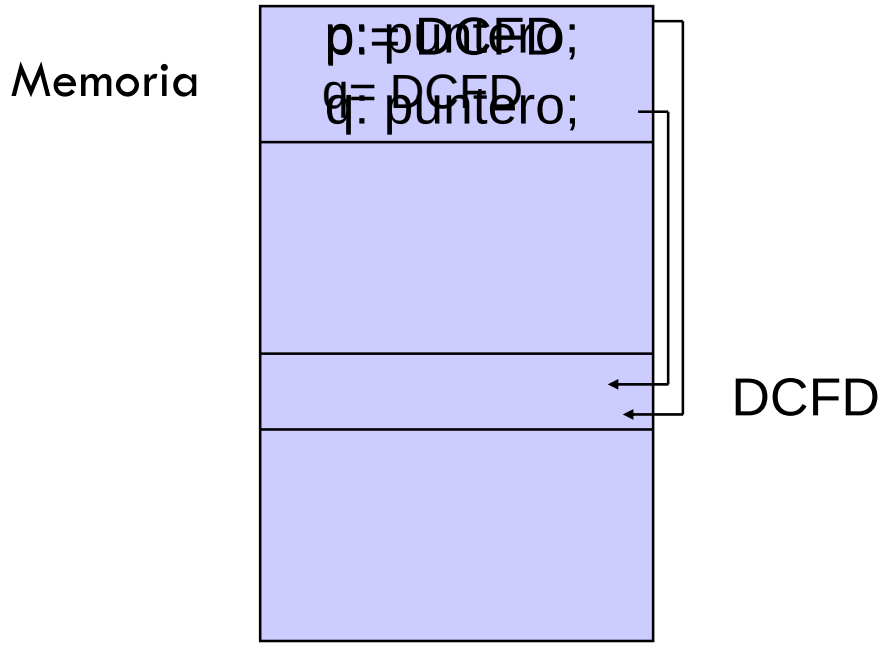


La operación p:=nil sólo elimina “la conexión” que existe entre la variable puntero y su dirección; la memoria sigue ocupada y no puede ser reasignada, pero tampoco puede ser accedida.

PUNTEROS - ASIGNACION

Asignar la dirección de otro puntero

```
Program tres;
Type
  puntero = ^integer;
Var
 p,q:puntero;
Begin
  new (p);
  …..
   q:= p;
End.
```



PUNTEROS - ASIGNACION

¿Cómo queda el gráfico de la memoria?

```
Program tres;
Type
  puntero = ^integer;
Var
 p,q:puntero;
```

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

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

Operación COMPARACIÓN DE PUNTEROS

PUNTEROS – Comparación

```
Begin
  new (p);
  …
  if (p = nil) then ….

  if (p = q) then


End.
```

Compara si la dirección de un puntero es nil

Compara si las direcciones de dos punteros son iguales

Operaciones NO PERMITIDAS

PUNTEROS – Operaciones NO permitidas

No se permite leer una variable de tipo puntero read(p);

No se permite escribir una variable de tipo puntero write(p);

No se permite asignar el contenido a una variable puntero de manera manual p:= AABC;

No se permite comparar por menor o mayor la dirección de una variable puntero if (p > ADBC)

if (p > q)

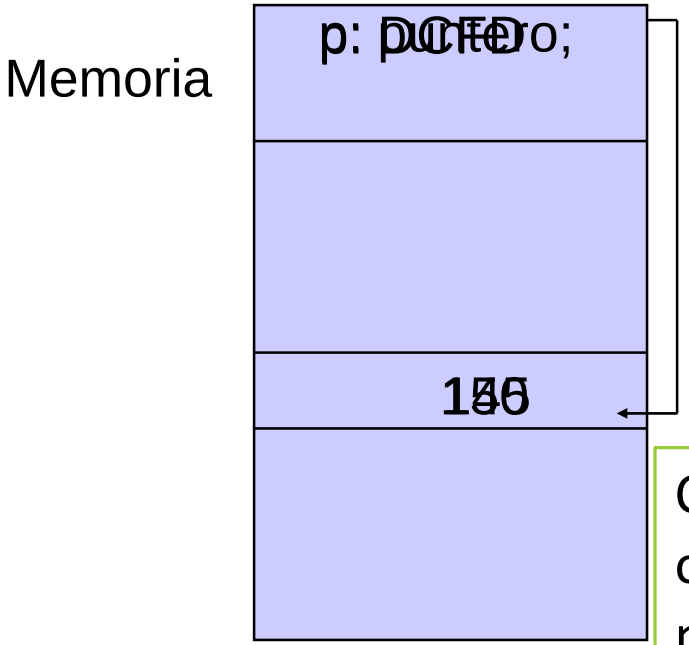
Operación ASIGNACIÓN DE VARIABLES DINÁMICAS

PUNTEROS – ASIGNACION- CONTENIDO

Para acceder al contenido de una variable de tipo puntero, utilizo el símbolo ^.

```
Program tres;
Type
  puntero = ^integer;
Var
 p:puntero;

Begin
  new (p);
  …..
  p^:= 145;
  p^:= p^ + 5;
End.
```



Cuáles son las operaciones permitidas para p^?

PUNTEROS

Qué imprime el programa?

```
Program tres;
Type
  puntero = ^integer;
Var
 p,q:puntero;
```

```
Begin
  new (p);
  .....
  new (q);
  q^:= 34;
   p^:=0;
  write(q^);
  write(p^);
End.
```

PUNTEROS

Qué imprime el programa?

```
Program tres;
Type
  puntero = ^integer;
Var
 p,q:puntero;
```

```
Begin
  new (p);
  p^:= 145;
  write (p^);
  q:=p;
  q^:= q^+10;
  write (p^);      write (q^);
  dispose (q);
  write (p^);     write (q^);
End.
```

EJERCICIOS


S
iable que
irección de otra variable.
Apuntador
Variable
dirección
valor
Se puede acceder al objeto
"indirectamente"

PUNTEROS – Ejercicio

¿Cuánta memoria es ocupada por PE?

Tamaño de las variables:
Char = 1byte
Integer = 2 bytes
Real = 6 bytes
Boolean = 1 byte
String = cantidad de caracteres + 1
Registro = la suma de lo que ocupa c/ campo
Puntero = 4 bytes

```
Program cinco;
Type
   Estudiante = record
                    Nombre : string [20];
                    Calificacion : integer;
                End;
   PunEstudiante = ^ Estudiante;

Var
   PE: PunEstudiante;
```

4 bytes

PUNTEROS – Ejercicio

```
Program cinco;
Type
        Estudiante = record
                        Nombre : string [20];
                        Calificacion : integer;
                      end;
   PunEstudiante = ^ Estudiante;
Var
  PE: PunEstudiante;
Begin
  new (PE);
```

¿Cuánta memoria es ocupada por PE? Solo 4 bytes

4 Bytes + (21 + 2) = 27 Bytes – Se reservan 23 bytes en tiempo de ejecución, en la heap.

PUNTEROS – Ejercicio

Analicemos el siguiente ejemplo:

```
Type
    cadena = string [255];
    pcadena = ^ cadena;

    pun1 = array[1…3000] of cadena;
   pun2 = array[1..3000] of pcadena;
Var
    p1: pun1;     3000 x 256 bytes
   pa: pun2;      3000 x 4 bytes
```

¿Cuánta memoria ocupa p1?

¿Cuánta memoria ocupa pa?

Cargue las dos estructuras.
¿Cuánto ocupan ahora?

Ya podemos realizar la práctica de Alocación dinámica