

PROGRAMACIÓN

Unidad 6 - Parte 2: Punteros y arreglos.

Recordemos un poco...

Puntero

```
graph LR; P[Puntero] --> D1[Es una variable que contiene la dirección de memoria de otra variable]; P --> D2[Declaración de Punteros: tipo *nombre-vble-puntero]; P --> D3[* Operador de indirección]; P --> D4[& Operador de dirección]; P --> D5[Inicialización: punt_num = &numero;];
```

Es una variable que contiene la dirección de memoria de otra variable

Declaración de Punteros: **tipo *nombre-vble-puntero**

***** Operador de indirección

& Operador de dirección

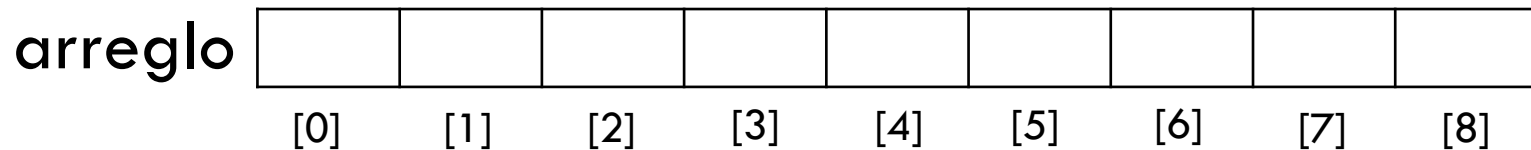
Inicialización: `punt_num = №`

Punteros y Arreglos

- En C existe una fuerte relación entre arreglos y punteros.
- Cualquier operación que pueda lograrse por indexación de un arreglo, también puede realizarse con punteros.
- Las versiones con punteros son más rápidas.

Declaraciones

```
int arreglo[9]
```



```
int *p_arreglo;
```

```
p_arreglo = &arreglo[0]; // asignación
```

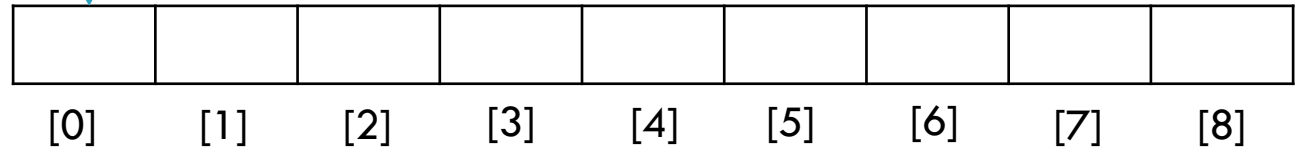
Asignación

p_arreglo



p_arreglo apunte al **elemento 0** del
arreglo

arreglo



Sabemos que, el nombre de un arreglo es sinónimo a la dirección del primer elemento del arreglo, por lo tanto la siguiente asignación es válida:

p_arreglo = arreglo;

Asignación

```
int aux;
```

```
int arreglo[9];
```

```
int *p_arreglo;
```

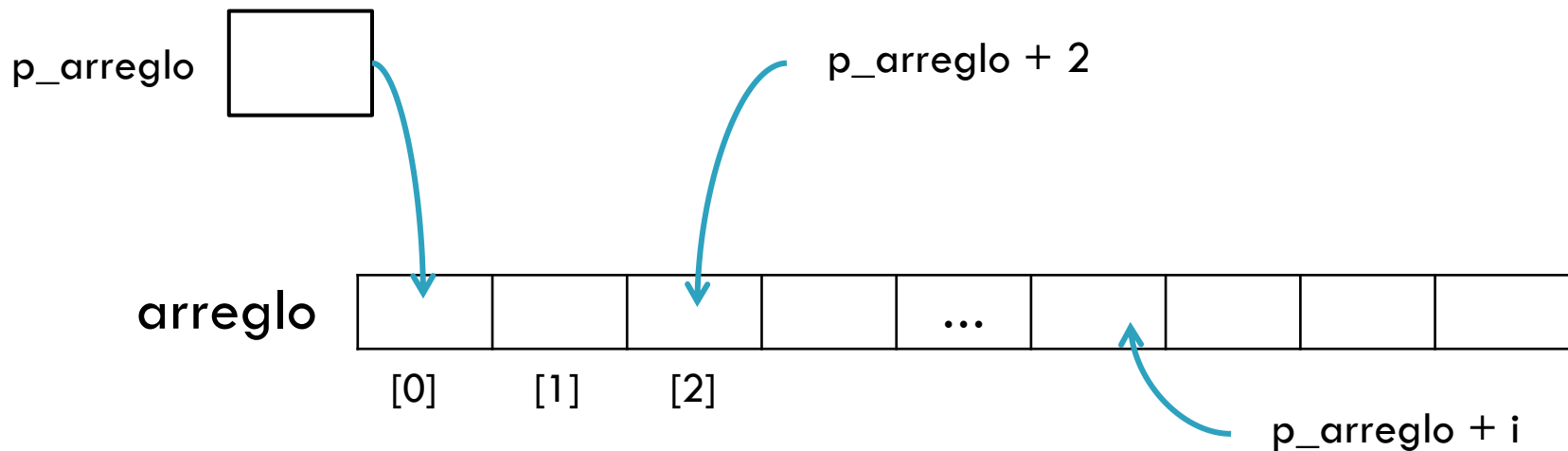
```
p_arreglo = arreglo;
```

```
aux = *p_arreglo; // copia el contenido del arreglo[0] en aux
```

arreglo	1	1	2	3	5	7	12	19	31
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]

En general

Si $p_arreglo$ apunta a un elemento en particular del arreglo,
 $p_arreglo+1$ apunta al siguiente elemento y $p_arreglo+i$ apunta i
elementos después de la dirección de $arreglo[0]$



Ejemplo

```
1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      char cadena[80];
7      int i;
8
9      puts("Introducir una cadena en mayuscula");
10     gets(cadena);
11
12     puts("Esta es la cadena en miniscula");
13
14     for(i=0; cadena[i]!='\0'; i++)
15     {
16         printf("%c", tolower(cadena[i]));
17     }
18
19     return 0;
20 }
21
```

ARREGLOS

```
1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      char cadena[80], *p_cadena;
7
8      puts("Introducir una cadena en mayuscula");
9      gets(cadena);
10
11     puts("Esta es la cadena en miniscula");
12     p_cadena = cadena;
13
14     while(*p_cadena != '\0')
15     {
16         printf("%c", tolower(*p_cadena) );
17         p_cadena++;
18     }
19
20     return 0;
21 }
```

PUNTEROS

Punteros

La velocidad es un bien preciado en programación, se debe saber que entre ambas versiones, la de punteros es más rápida.

Existe una diferencia muy importante entre nombre del arreglo y un apuntador.

Un puntero es una variable, es lícito escribir:

```
p_cadena = cadena ;
```

```
p_cadena ++;
```

En tanto: un nombre de arreglo NO es una variable, por lo cual no es correcto escribir:

```
cadena++;
```

```
cadena = punt;
```

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    printf("Hasta la próxima clase!!\n");
```

```
    return 0;
```

```
}
```