El lenguaje C. El tipo Puntero

Luis Llopis, 2024.
Dpto. Lenguajes y Ciencias de la Computación.
University of Málaga

UNIVERSIDAD DE MÁLAGA | uma.es

¿Por qué es importante?

Poder y Flexibilidad

Código eficiente y flexible

¿Por qué es importante?

1. **Acceso directo a la memoria**
2. **Manipulación eficiente de arrays y cadenas de caracteres**
3. **Paso por referencia**
4. **Creación de estructuras de datos complejas**
5. **Gestión dinámica de la memoria**

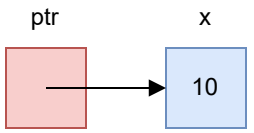
Declaración

- Declaración de una variable de tipo Puntero

```
int *ptr;
int * ptr1;
int* ptr2;
```

- Operador de dirección **&**
  - Devuelve la dirección en memoria de una variable
  - Es un operador unario

```
int *ptr;
int x = 10;
ptr = &x;
```



Declaración y uso

- Operador de indirección *
  - Devuelve el contenido de la zona de memoria a la que apunta un puntero
  - Es un operador unario
  - No confundir con la declaración de un puntero o con el operador multiplicación

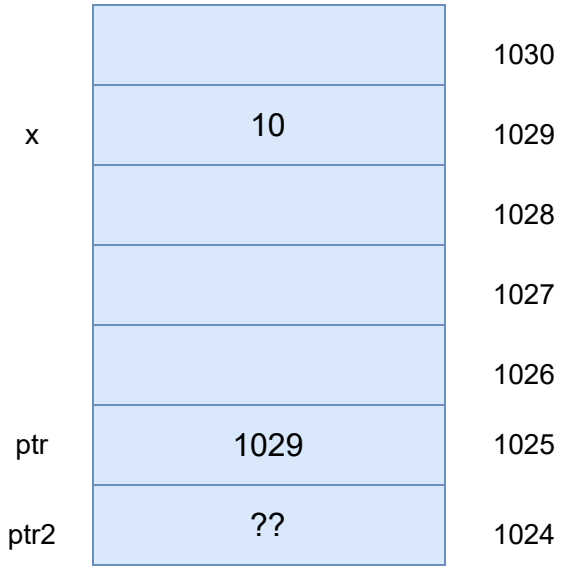
```
printf(“valor = %d %d”, x, *ptr);
```

Declaración y uso

```
int x = 10;
int *ptr;

ptr = &x;

double *ptr2;
```



Punteros

- Define una variable entera.
- Define un puntero
- Almacena la dirección de la variable entera anterior.
- Accede al contenido del puntero (a lo que apunta).
    - A esto se denomina indirección o desreferenciación del puntero, es decir, acceder al valor al que apunta la dirección del puntero.
- Muestra los valores y las direcciones de memoria (modificador *%p*)

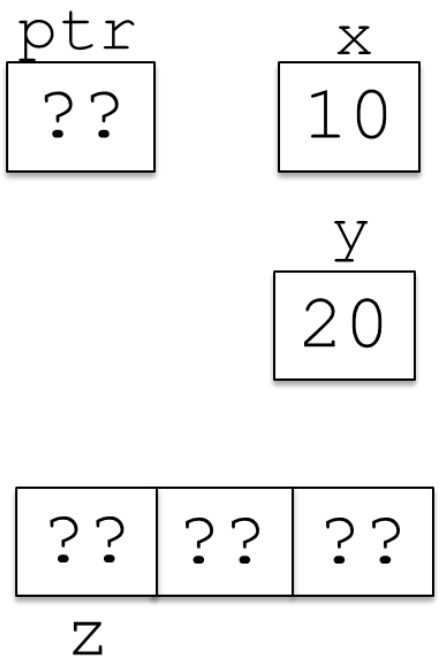
El tipo puntero

- Un puntero tiene la dirección donde se encuentra un valor de un tipo.
- ¿De qué tipo puede ser esa variable ?

¡¡¡¡DE CUALQUIERA!!!!

Ejemplo (1/3)

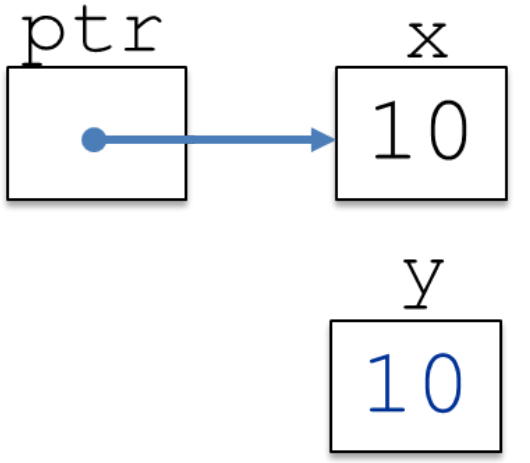
```
int x = 10;
int y = 20;
int z[3];
int *ptr;   // ptr es un puntero a un entero
```

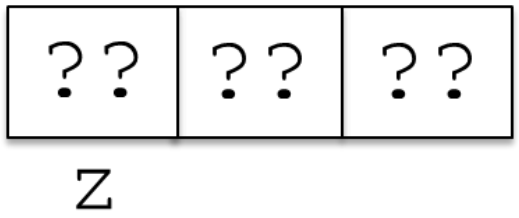


48

Ejemplo (2/3)

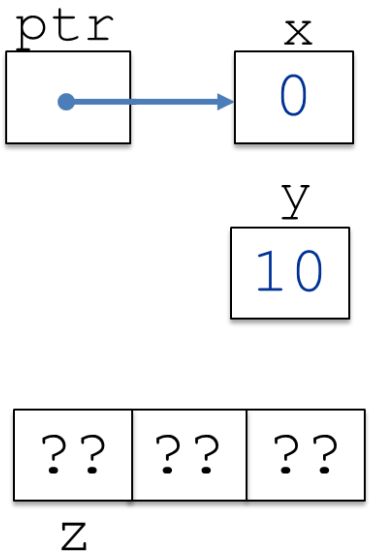
```
ptr = &x;    // ptr apunta a x
y = *ptr;    // *ptr accede al valor apuntado por ptr
             //   (y, vale 10)
```



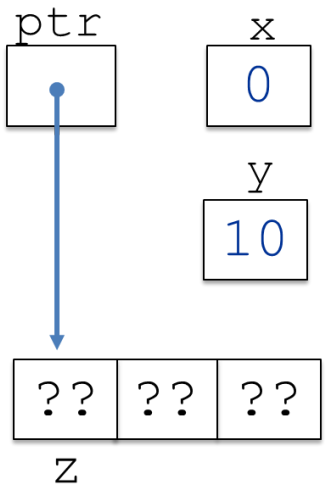


Ejemplo (3/3)

```
*ptr = 0;     // el valor de x es 0
```




```
ptr = &z[0]; // ptr apunta a z[0]
```



El Tipo Puntero y su utilidad en Procedimientos y Funciones

Procedimientos y Funciones

- C define funciones que pueden:
  - Devolver “nada” (void) -> Procedimientos
  - Devolver valores de algún tipo (incluído punteros)
- Argumentos de las funciones
  - Todos se **pasan por valor** pero….
  - **Es posible pasarlos por "referencia"**.
    - En la llamada se debe proporcionar la dirección de la variable
    - En la definición formal el parámetro debe ser declarado como puntero a una variable del tipo correspondiente.

Parámetros por valor y referencia

INCORRECTO

```
swap (a, b); //llamada
…
void swap(int x, int y) {
  int temp;
  temp = x;
  x = y;
  y = temp;
}
```

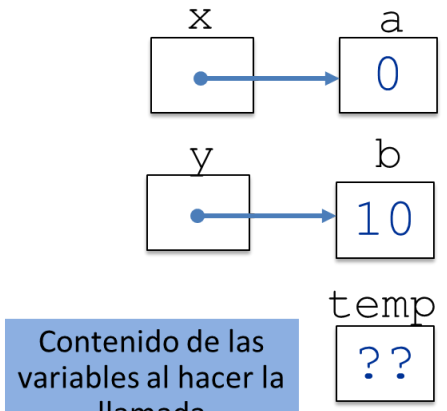
CORRECTO

```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
  int temp;
  temp = *x;
  *x = *y;
  *y = temp;
}
```

- Prueba los ejemplos anteriores

Paso por referencia

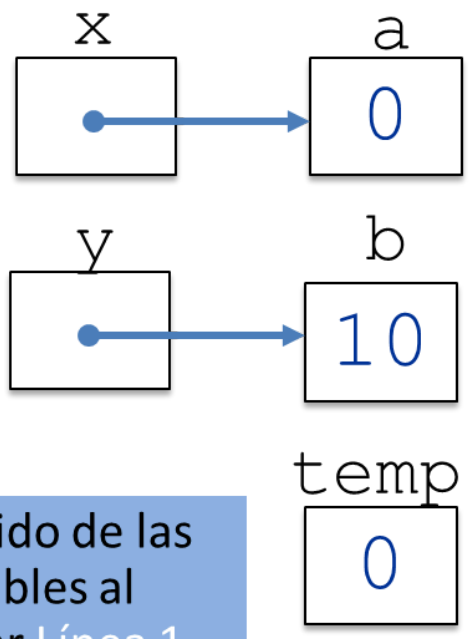
```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
  int temp;
  temp = *x;
  *x = *y;
  *y = temp;
}
```



Paso por Referencia

```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
   int temp;
   temp = *x;
   *x = *y;
   *y = temp;
}
```

Línea 1

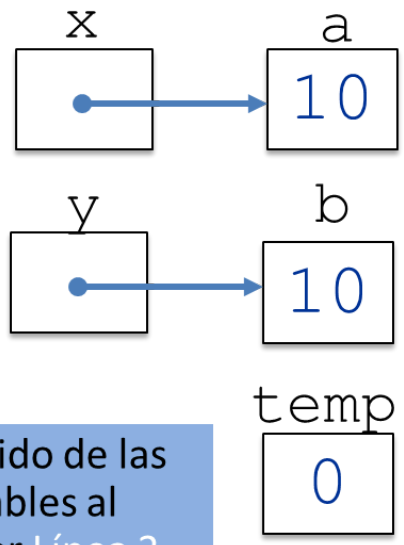


Contenido de las variables al ejecutar Línea 1

Paso por Referencia

```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
  int temp;
  temp = *x;
  *x = *y;
  *y = temp;
}
```

Línea 2

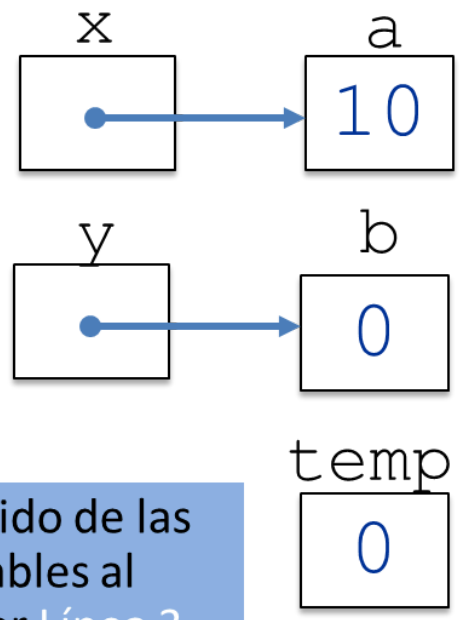


Contenido de las variables al ejecutar Línea 2

Paso por Referencia

```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
   int temp;
   temp = *x;
   *x = *y;
   *y = temp;
}
```

Línea 3



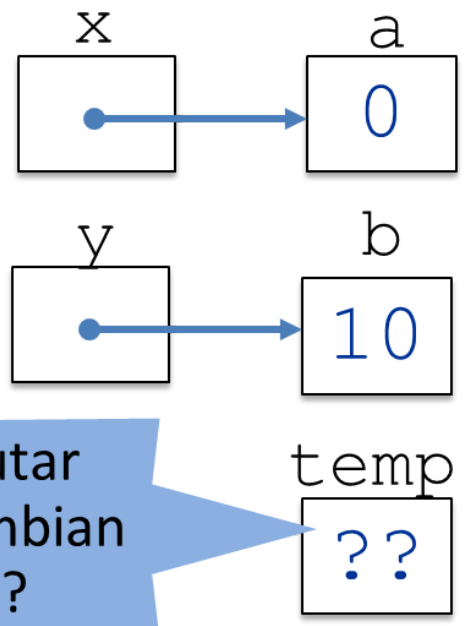
Contenido de las variables al ejecutar Línea 3

Paso por Referencia

```
swap (&a, &b); //llamada
…
void swap(int* x, int* y) {
   int * temp;
   temp = x;
   x = y;
   y = temp;
}
```

Incorrecto

¿Qué pasaría al ejecutar este otro código? ¿Cambian los valores de a y b?



Paso por valor/referencia

Realiza un programa que suma dos números y modifica uno de ellos por referencia.

1. Asignar dos números a dos variables en el *main*
2. Definir una función que suma dos números y modifica el segundo de ellos para devolver la suma (paso por referencia).
3. Mostrar en el main los valores de las dos variables

El tipo Puntero

- Sabemos definir un tipo puntero
- Sabemos que el tipo puntero sirve para pasar parametros por referencia
- Hemos comentado que una varialbe puntero puede apuntar a cualquier tipo
    - Solo hemos visto apuntando a tipos simples

El tipo Puntero

- Apuntar a tipos simples no aporta mucho con respecto a definir variables estáticas.

Hay otras formas de aprovechar la flexibilidad de los Punteros