UNIVERSIDAD NACIONAL EXPERIMENTAL DEL TÁCHIRA

VICERRECTORADO ACADÉMICO

DECANATO DE DOCENCIA

DEPARTAMENTO DE ING. EN INFORMÁTICA

UNIDAD CURRICULAR: COMPUTACIÓN I

(Código: 0415102T)

Clase No: 10

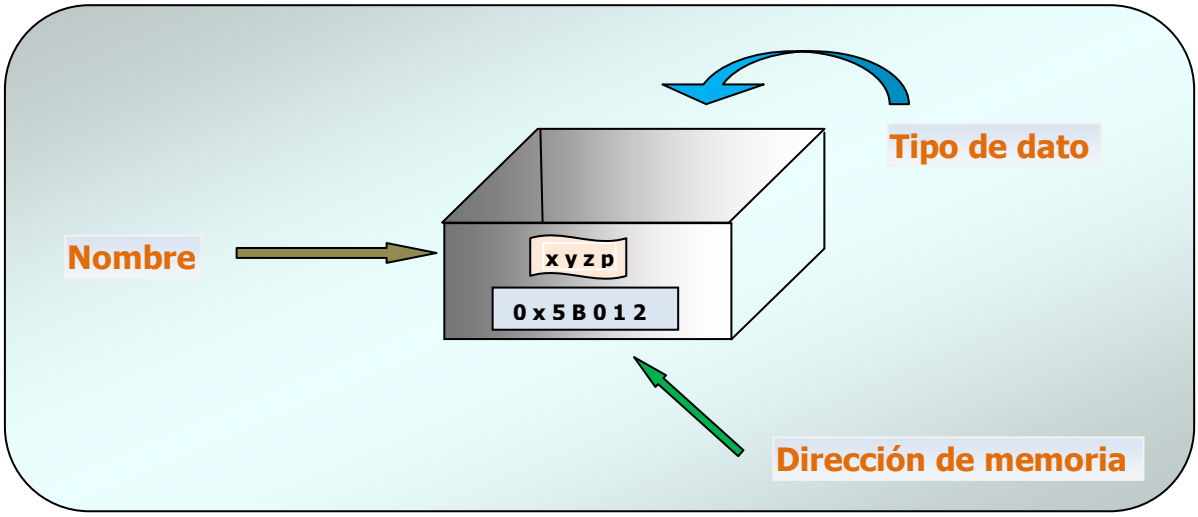
Unidad VI: Punteros en Lenguaje C

Profesor: Armando Carrero

UNIDAD No. VI
PUNTEROS EN LENGUAJE C

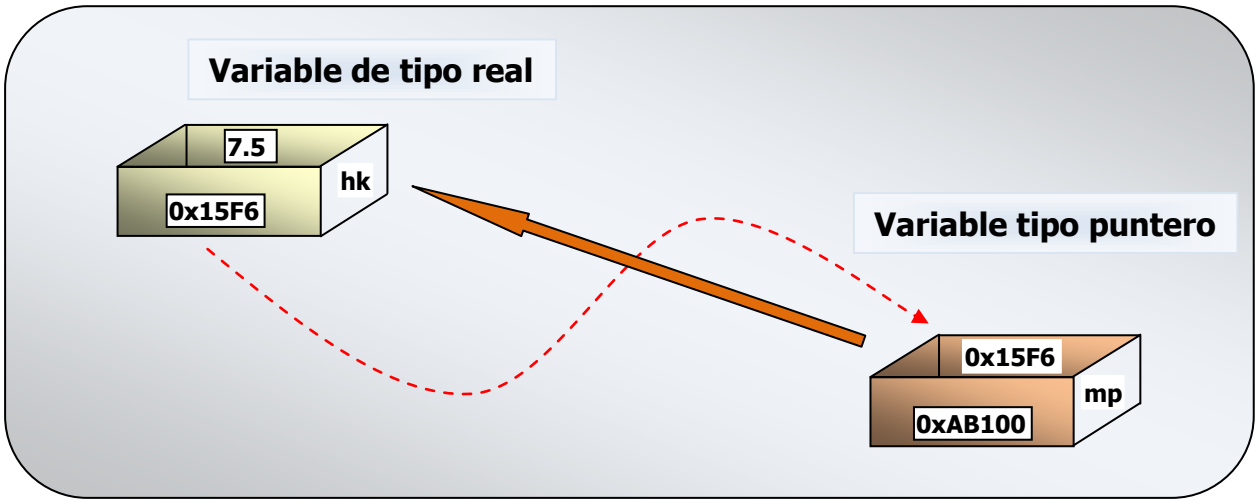
¿Qué es un Puntero? : Es una variable, y como se conoce, las variables sirven para almacenar en ellas, valores; solo que en éste caso no se trata de un valor como los que se han tratado hasta ahora, es decir: un número entero, un número real, un carácter, una cadena de caracteres o un valor lógico.

Para concebir claramente, lo que puede almacenarse en una variable tipo puntero, se debe recordar lo siguiente: Toda variable declarada, se caracteriza por adquirir tres atributos o características: Un **nombre**, un **tipo de dato** y **una dirección de memoria**, que se ilustran a continuación:



Luego, en una variable tipo puntero ***se puede almacenar o guardar la dirección de memoria de otra variable***. La dirección es asignada por la computadora en la cual se ejecuta el programa, y puede ser diferente de una maquina a otra. La dirección es un número o código en sistema hexadecimal, que usa dígitos del 0 al 9 y las letras de la A a la F (la A representa el 10, la B el 11, la C el 12, la D el 13, la E el 14 y la F el 15).

Para ilustrar aun más, supóngase la variable ***hk*** de tipo real con el valor 7.5 almacenado en ella; y la variable ***mp*** de tipo puntero, que se pueden representar así:

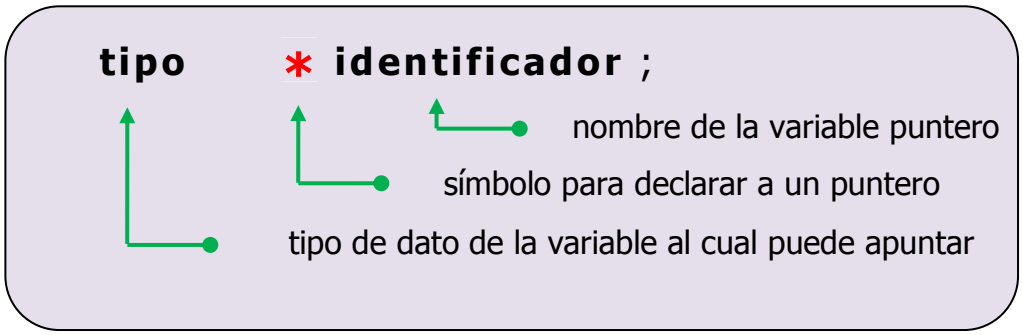


La variable tipo puntero ***mp*** almacena la dirección de memoria de la variable ***hk***, es decir el valor **0x15F6,** luego se dice que la variable ***mp*** "**apunta**" a la variable ***hk*** **.**

DECALARACION DE UNA VARIABLE TIPO PUNTERO

De manera semejante a las variables ya estudiadas, las variables tipo puntero deben ser declaradas antes de utilizarse, para que el compilador conozca el tipo de dato al cual puede apuntar, para ello debe atenderse al siguiente formato:

Formato:



Ejemplo:

```
int     * val  ;
float   * pro  ;
char    * est  ;
long    * diz  ;
```

Por lo tanto, ***cuando en la sección de declaraciones, aparezca el símbolo * precediendo a una variable, ésta será una variable tipo puntero*** con capacidad para almacenar la posición de memoria de otra variable, del tipo de dato indicada en la declaración.

Es decir que en la declaración anterior, se debe entender lo siguiente:

*** val** mediante la variable **val** se puede apuntar a otra variable de tipo entero;
*** pro** **pro** es un puntero a un tipo de dato real;
*** est** la variable **est** serviría para apuntar a otra variable de tipo caracter ;
*** diz** **diz** puede guardar la dirección de otra variable entero largo.

Es necesario tener claro que una variable declarada tipo puntero, por ejemplo así: int *** val** ; no debe interpretarse que en ella (**val**) se puede almacenar un valor de tipo entero, lo correcto es que en ella se puede almacenar la dirección de otra variable de tipo entero.

OPERADORES CON PUNTEROS

Existen dos operadores para el manejo de punteros:

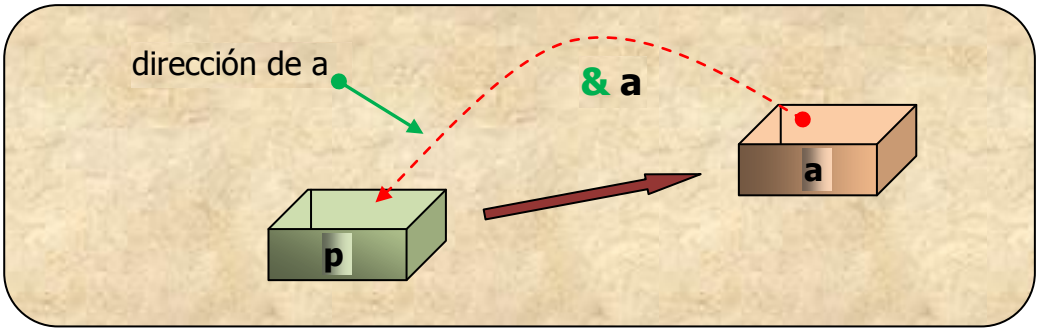
Operador dirección **&** : Devuelve la dirección de una variable
Operador indirección ***** : Permite acceder al valor de la variable apuntada

Así, mediante el **operador &**, se referencia la dirección de una variable para asignársela a la variable tipo puntero, ejemplo:

```
int   a ;        /*  declaración de la variable a de tipo entero  */
int  *p ;        /*  declaración de la variable p de tipo puntero a un entero */

p = & a ;        /* asignación de la dirección de la variable a al puntero p  */
```

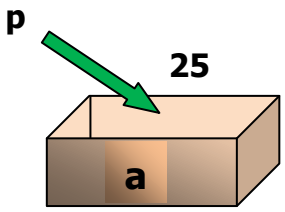
Gráficamente se puede representar así:



Mediante la asignación **p = & a ;** la variable **p** "apunta" a la variable **a**, es decir, la dirección de la variable **a**, cuyo valor no importa cuál sea, se almacena en la variable **p.**

Por otra parte, mediante el **operador** *, se puede acceder al valor de la variable **a,** usando el puntero **p**, bien sea para asignarle un valor o para usar su valor en cualquier operación o proceso, ejemplo:

```
*p  = 25 ;     /* se asigna el valor 25 a la indirección de p, es decir, a la variable a */
```




```
int    b ;          /*  declaración de la variable b de tipo entero  */

b = 5  +  *p  ;     /*  se asigna a la variable b:   5 + el valor de la variable a la que */
                    /* apunta p, es decir 30   (5 + 25)  */
```

Una utilidad fundamental que se le da a los punteros, es para que una función pueda retornar varios valores, ya que con la instrucción return solo puede devolver un único valor. Este caso se presenta en ejemplo a continuación:

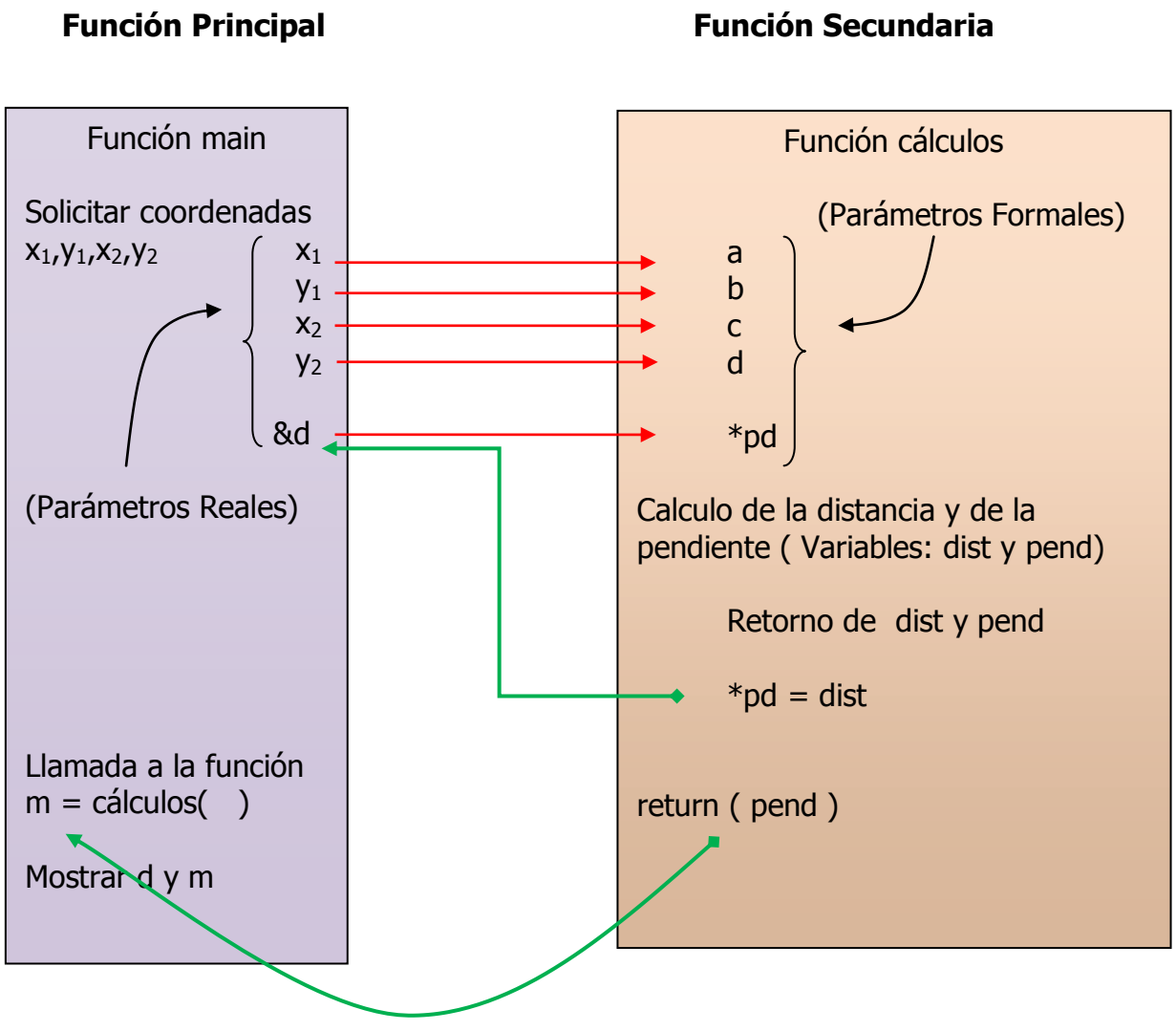
PROBLEMA SOBRE USO DE PUNTEROS

Conocidas las coordenadas, en centímetros, de dos puntos: $P_1(x_1,y_1)$ y $P_2(x_2,y_2)$; diseñe un programa en lenguaje C, que utilice una función secundaria con parámetros, que sea ejecutada mediante una asignación y que retorne:

a) la distancia entre los puntos P_1 y P_2 $\quad d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

b) la pendiente de la recta que pasa por los puntos P_1 y P_2 $\quad m = \frac{y_2 - y_1}{x_2 - x_1}$

Planificación



```
//Programa para calcular la distancia entre dos puntos y la pendiente de la recta que pasa por ellos

//Archivos de Cabecera

#include<stdio.h>
#include<math.h>
#include<stdlib.h>

//Prototipo de la función calculos (declaración de la función cálculos)

float  calculos ( int x1, int y1, int x2, int y2, float *pd ) ;

        //Función Principal main

int   main ( )
  {
    //declaración de variables locales

     int    x1, y1, x2, y2 ;
     float  m , d ;

     printf("\n Ingrese las coordenadas x1 y y1, respectivamente,  del punto P1:   ") ;
     scanf ("%d%d" , &x1, &y1 ) ;

     printf ("\n Ingrese las coordenadas x2 y y2, respectivamente,  del punto P2:   ") ;
     scanf ("%d%d" , &x2, &y2 ) ;

     // Llamada a la función calculos
      m =  calculos  (x1 , y1 , x2 ,  y2 , &d ) ;

     printf("\n La distancia entre los puntos P1 y P2 es  = %8.2f cm " , d) ;
     printf("\n La pendiente de la recta que pasa por los puntos P1 y P2 es  \n= %8.2f " , m) ;
     system ("pause") ;
     return 0 ;

  }     //Fin de la función principal main

        //Función Secundaria calculos

float  calculos ( int a, int b, int c, int d, float *pd )  // Definición de la función cálculos)
   {
     //declaración de variables locales
     float   dis, pend ;
     dis = sqrt ( pow ( c – a , 2) + pow ( d – b , 2) ) ;
     pend  = ( d – b ) / ( c – a ) ;
     *pd = dis ;
     return  ( pend ) ;

    }    //Fin de la función calculos
```