

Puntero a Puntero

Concepto

Como vimos en módulos anteriores un puntero es una variable que apunta a otra, fácilmente podemos deducir que pueden existir punteros a punteros, y a su vez los segundos pueden apuntar a punteros, y así sucesivamente. Estos punteros se declaran colocando tantos asteriscos ('*') como sea necesario. Para especificar que una variable es un puntero a un puntero, la sintaxis que debemos utilizar es la siguiente:

Tipo **variable;

donde **Tipo** especifica el tipo de dato apuntado después de una doble indirección. ¿Pero qué es una doble indirección? Veamos el siguiente ejemplo:

```
int a, *p, **pp;  
a=10;  
p=&a; //Puntero que apunta al dato  
pp=&p; //Puntero que apunta al puntero que a la vez apunta al dato
```

Decimos que p es una variable con un solo *nivel de indirección*, es decir, **a través de p no accedemos directamente al dato, sino a la dirección que indica donde está el dato**. Haciendo un razonamiento similar, diremos que pp es una variable con dos niveles de indirección.

Veamos el siguiente código:

```

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

int main()
{
    int *p, a;
    int **q;
    a=10;
    p=&a;
    q=&p;
    **q=*p;
    printf ("%d %d", *p, **q);
    a=20;
    **q=*(&a);
    printf ("\n%d %d", *p, **q);
    return 0;
}

```

Se define una variable q como doble puntero a un entero

El puntero p toma la dirección de la variable a y el puntero q toma el valor del puntero p, es decir 10

El puntero q toma el contenido de la variable a. Esto es, la dirección de memoria donde se almacenó a (&a), *(&a) toma el contenido de a.

*p mostrará 20 porque p apunta a la variable a y esta se actualizó.
**q mostrará 20

A partir de ahora en muchos códigos utilizaremos punteros a punteros.