**Almacenando una dirección en un puntero**

* Cada variable tiene una dirección.
* Aun cuando no conozcamos la dirección de una variable, podemos almacenar esa dirección en un puntero.
* Supongamos que tenemos una variable entera llamada edad. Ära declarar un puntero que almacene la dirección de la variable edad escribimos la siguiente sentencia:

**Int \*pEdad = NULL;**

NULL es una constante declarada por el sistema estándar que vale 0, que apunta a ninguna dirección de memoria. (se escribe mayúscula y por eso es constante).

**Declaración de un puntero**

* Cuando declaramos un puntero es conveniente que este apunte inmediatamente a alguna dirección.



* En el ejemplo anterior lo inicializamos a NULL.
* NULL es una constante cuyo valor es 0.
* Un puntero inicializado a NULL se llama puntero nulo.

**Nullptr**

* La mayoría de los compiladores nuevos tienen una constante especial llamada nullptr que representa un puntero nulo, por lo que es recomendable utilizarla en vez de NULL.

**Int \*pEdad = nullptr;**

**Asignación del puntero**

**int edad = 25; //declaración de la variable  
int \*pEdad = nullptr; //Declara el puntero  
pEdad = &edad; //Pone la dirección de edad en pEdad**

**Operador de induración**

* El operador de indireccion es el \*
* Se le conoce también como operador de desreferenciacion.
* Cuando un puntero es desreferenciado, se obtiene el valor almacenado en la dirección a la que apunta en puntero.

Se obtiene el valor que hay almacenado en ese lugar

**Indireccion**

* El puntero provee acceso indirecto al valor de la variable cuya direccion el almacena.
* Para asignar el valor de la ariable sueldo a una nueva variable otroSueldo por medio del punto pSueldo. Se haria lo siguiente:

float sueldo = 25654.34; //Crea la variable dueldo  
float \*pSueldo = &sueldo; //pSueldo apunta a la direccion de sueldo  
  
float otroSueldo; //Crea otra variable  
otroSueldo = \*pSueldo; //Asigna el valor en pSueldo (25654.34) a otroSueldo

* *(\*)Aterisco me trae el valor de la direccion del puntero*

*Da lo que tiene el valor de la variable a donde el apunta.*

* *(&) dame la direccion de la variable que lleve & delante*

*ejemplo:*

*Int edad = 20;*

*Cout <<&edad <<endl;*

*Imprime la direccion donde esta ese valor. Ejemplo*

**Confusiones**

* Es muy importante distinguir entre:
* Un puntero
* La direccion que el puntero almacena.
* El valor almacenado en la direccion a la que apunta el puntero.
* Ya que no tener este bien claro es la fuente de la mayoria de las confusiones con los punteros.

**Manipulación de datos con puntero.**

* Una vez que se asigna la dirección de una variable a un puntero, podemos utilizar el puntero para acceder a los datos que se encuentran en la variable.
* El siguiente programa muestra como se utilizan los punteros para manipular los valores de las variables.

**Contenido de los puntero**

* Los punteros permiten manipular direccion sin siquiera conocer los valores almacenados en ellas.
* El siguiente programa muestra el contenido de los punteros.

**Para que se utilizan los punteros**

* Manejar datos en los monticulos (heap)
* Acceder a datos y funciones muembro de las clases.
* Para pasar variables por refrencia a las funciones.

**Ejercicio**

* Modifique el programa RevisaPuntero.cpp paa que multiplique tuEdad y \*pEdad y almacene el resultado en una nueva variable. Desplique esa variable. Piense como el compilador puede notar la diferencia entre operador \* para la multiplicacion y el operaro \* ara desreferenciar pEdad.
* Modifique a RevisaPuntero.cpp para usar el puntero desreferenciado \*pEdad para cabiar el conenido de miEdad o de tuEdad.

El operador de induración (\*) dice el valor que está en esa dirección.