Laboratorio de Computación II

Clase 02:
Punteros

Punteros

Un puntero es un tipo de variable que puede almacenar la dirección de memoria de otra variable. Para hacerlo hay que garantizar las siguientes reglas:

- **Un puntero debe declararse de un tipo de dato en concreto**
- **Un puntero debe apuntar a una variable de su mismo tipo (salvo punteros void)**

Operadores de dirección e indirección

Para trabajar con punteros es necesario hacer uso de dos operadores especiales.

Operador de dirección: &
Permite obtener la dirección de memoria de una variable.

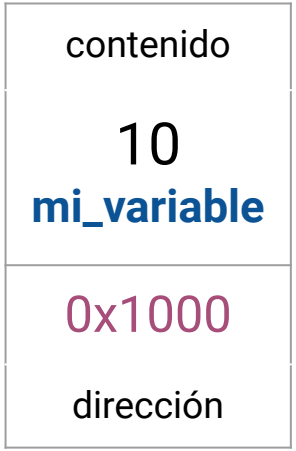
Operador de indirección *
Permite acceder al contenido de una dirección de memoria.

Operador de dirección

Permite obtener la dirección de memoria a partir de una variable

Ejemplo

```
int mi_variable = 10;
cout << mi_variable;
cout << &mi_variable;
```



Operador de indirección

Permite acceder al contenido a partir de una dirección de memoria

Ejemplo

```
int mi_variable = 10;
int *mi_puntero;
mi_puntero = &mivariable;
cout << *mi_puntero;
```

contenido	contenido
10 **mi_variable**	0x1000 **mi_puntero**
0x1000	0x2000
dirección	dirección

Ventajas

- **Enviando la dirección de una variable a una función, permite que la misma modifique su contenido de forma permanente.**
- **Necesarios para hacer uso de Asignación dinámica de memoria.**
- **Al usarlos bien hacen felices a los profesores de Laboratorio II.**

Punteros y vectores

Un puntero puede apuntar a la dirección de inicio de un vector de su mismo tipo y ser de utilidad para acceder a todas sus posiciones.

```
int mi_vector[10]={};
int *mi_puntero;
mi_puntero = mi_vector;
mi_puntero[4] = 100;
```

Recordar que el nombre de un vector es un **puntero constante a la dirección donde inicia el vector**.

¿Y el operador de indirección? El uso de corchetes indirecciona un puntero en la posición indicada. Es equivalente a: ***(mi_puntero+4) = 100**

Aritmética de punteros

La notación *(mi_puntero+4) corresponde a la aritmética de punteros. Es decir, aplicar operaciones matemáticas a punteros.

```
vec   → &vec[0] → 0x100
vec+0 → &vec[0] → 0x100
vec+1 → &vec[1] → 0x10C
```

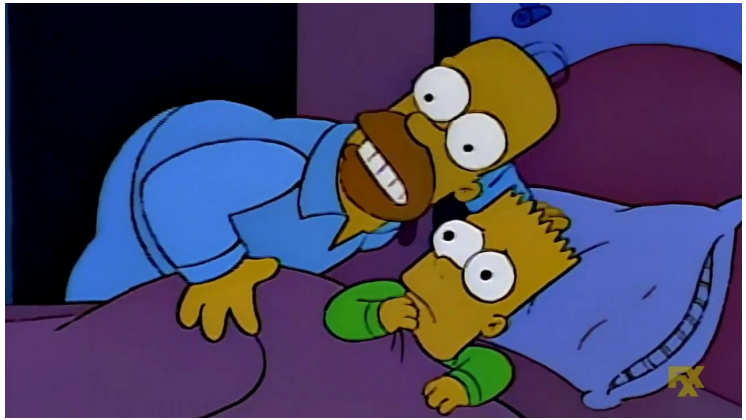
vec+N → &vec[N]

```
*(vec)   → *(&vec[0]) → 100
*(vec+0) → *(&vec[0]) → 100
*(vec+1) → *(&vec[1]) → 200
```

***(vec+N) → *(&vec[N]) → vec[N]**

Elemento	Contenido	Dirección
vec[0]	100	0x100
vec[1]	200	0x104
vec[2]	300	0x10C
vec[3]	400	0x110
vec[4]	500	0x114

Aritmética de punteros



No quiero asustarlos pero **también aplica a matrices** de N dimensiones.

```
*(*(vec+0)+0) → vec[0][0]

*(*(vec+3)+5) → vec[3][5]

*(*(vec+F)+C) → vec[F][C]
```

Actividad

Hacer una función que permita cargar un vector de 10 elementos y otra que permita obtener el máximo valor de un vector de 10 elementos. Usar aritmética de punteros.