Escuela Politécnica Superior

SEMINARIO-TALLER DE SOFTWARE

(STI-S)

UNIDAD 6. Programación con punteros

(Sesión 1)

Índice

Índice

Introducción

- Un puntero es un dato que contiene una **dirección de memoria.**

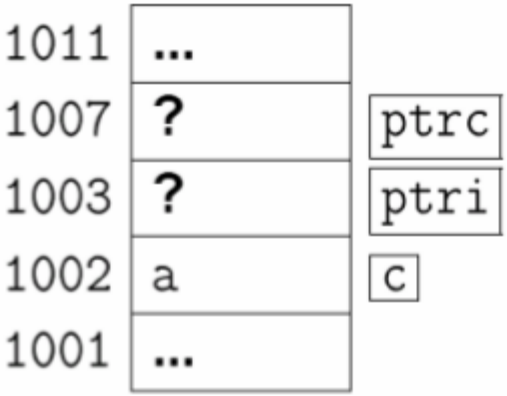
- **NOTA:** Existe una dirección especial que se representa por medio de la constante NULL.

- ¿Cómo se declara una variable puntero?

```
tipo_de_dato *nombre_variable_puntero;
```

Introducción

```
char c = ‘a’;
char * ptrc;
int * ptri;
```



- **NOTA:** Cuando se declara un puntero se reserva memoria para almacenar una dirección de memoria, NO PARA ALMACENAR EL DATO AL QUE APUNTA EL PUNTERO

Índice

El operador &

- Este operador, usado junto con el nombre de la variable puntero, se utiliza para **conocer la dirección de memoria** donde comienza la variable.

- Ejemplo:

```
int i = 5;
int *pointer;
pointer = &i;
```

Índice

El operador *

- El operador *, usado junto con el nombre de la variable puntero, se utiliza para **acceder al contenido** referenciado por la variable puntero.

- Ejemplo:

```
int i = 5;
int *pointer;
pointer = &i;
*pointer = 3;
```

Índice

El operador =

- Al igual que con variables normales, el operador = se utiliza para dar valor a las variables puntero.

- Una buena costumbre consiste en inicializar los punteros a NULL cuando la variable puntero es declarada:

```
int *pointer = NULL;
```

El operador =

- Para asignar un valor a una variable puntero podemos hacer lo siguiente:

1. Asignarle la dirección de memoria directamente:

```
int *pointer = 0x1F3CE00A;
```

2. O dándole la dirección de una variable del mismo tipo a la que apunta el puntero:

```
int c = 4;
int *pointer = &c;
```

3. U otro puntero del mismo tipo:

```
int c;
int *pointer = &c;
int *pointer2 = pointer;
```

4. Usando la función malloc (se verá en un futuro)

Ejemplo

```c
#include<stdio.h>

void main(){
        int variable = 1;
        int *pointer = &variable;

        printf("Acceso directo a variable=%d\n",variable);
        printf("Acceso indirecto a variable=%d\n", *pointer);

        printf("La direccion de variable=%p\n", &variable);
        printf("La dirección de pointer=%p\n", pointer);
}
```

```
Acceso directo a variable=1
Acceso indirecto a variable=1
La direccion de variable=0x7fffced4056c
La dirección de pointer=0x7fffced4056c
```

Índice

Errores comunes

```
int x = 10;
int * ptr1 = NULL;
double y = 0.5;
double * ptr2 = NULL;
…
ptr1 = &x;
ptr2 = &y;
ptr1 = ptr2; // ERROR
```

```
int * ptr1;
*ptr1 = 3; // ERROR
```

```
char c;
int * ptr = &c;
ptr = ‘a’; // ERROR
```

```
int * ptr1 = NULL;
*ptr1 = 3; // ERROR
```