

# PUNTEROS

PROGRAMACIÓN (HOMOLOGADA)

FACULTAD POLITECNICA



# Punteros

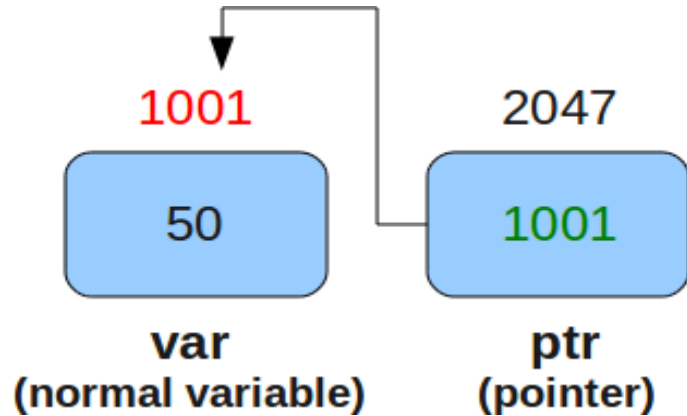
- Punteros y variables simples



# PUNTEROS DEFINICIÓN

Un **puntero** es una variable que contiene la dirección de memoria de otra variable

**No hay que confundir una dirección de memoria con el contenido de esa dirección de memoria**



Dirección		1502	1504	
1506	1508	25	...	...
...	...	...	...	...

La dirección de la variable x (&x) es 1502

El contenido de la variable x es 25

**(&)** Devuelve la dirección de memoria del operando.

**(\*)** Devuelve el valor almacenado en la dirección de memoria que determina el operando.



# PUNTEROS DEFINICIÓN

- El operador “&” da la dirección de un objeto(variable, entonces la sentencias es `p=&c;`
- Asigna al puntero *p* la dirección de la variable *c*. Entonces se dice que “*p*” apunta a “*c*”. El operador “&” sólo se aplica a elementos almacenados en memoria(como variables o arreglos)
- El operador \* es el “desreferenciador ” o “indirección”. Al aplicarlo a un puntero, se accede al objeto al que apunta



# PUNTEROS Y VARIABLES SIMPLES

Una variable puntero se declara como todas las variables. Debe ser del mismo tipo que la variable apuntada. Su identificador va precedido de un asterisco (\*):

```
int *punt;
```

Es una variable puntero que apunta a variable que contiene un dato de tipo entero llamada punt.

```
char *car:
```

Es un puntero a variable de tipo carácter.

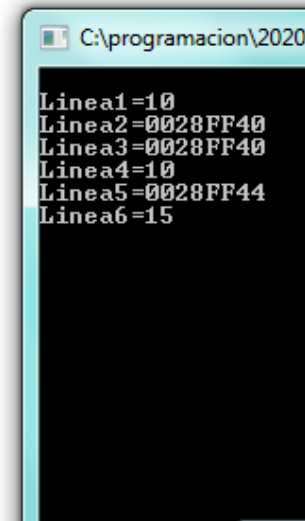


# PUNTEROS Y VARIABLES SIMPLES

<code>int x = 1, y=2</code>	<b>Declaración e inicialización de variables simples</b>
<code>int *ip;</code>	ip es un puntero a un int
<code>ip = &amp;x;</code>	ip apunta a la variable x
<code>y = *ip;</code>	ahora y es 1
<code>*ip = 0</code>	ahora y es 0

Observación: un puntero está restringido a apuntar a objetos de un tipo en específico (por eso se definen como un tipo de dato)

```
main() {  
    int *p;  
    int n=10;  
    p=&n;  
    printf("\nLinea1=%d", n);  
    printf("\nLinea2=%p", &n);  
    printf("\nLinea3=%p", p);  
    printf("\nLinea4=%d", *p);  
    printf("\nLinea5=%p", &p);  
    *p=15;  
    printf("\nLinea6=%d", n);  
    system("pause>null");  
}
```



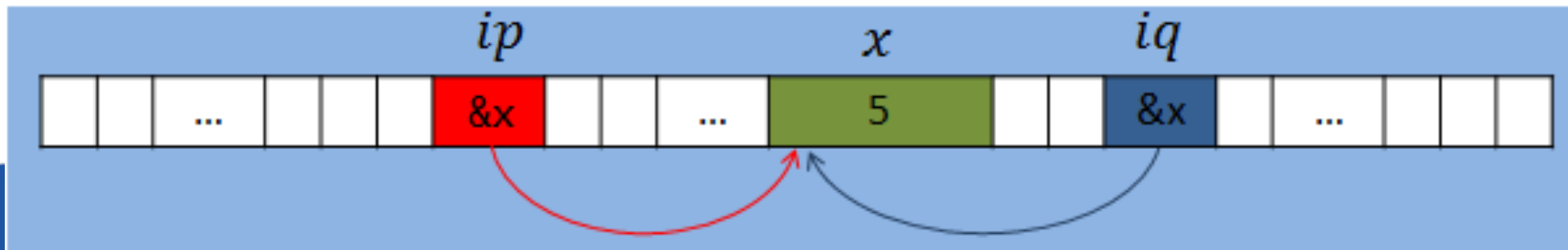
```
C:\programacion\2020  
Linea1=10  
Linea2=0028FF40  
Linea3=0028FF40  
Linea4=10  
Linea5=0028FF44  
Linea6=15
```



# PUNTEROS Y VARIABLES SIMPLES

Como los punteros son variables, se pueden hacer operaciones sin “desreferenciar”. Por ejemplo:

```
int x = 5;  
int *ip, *iq;  
ip = &x;  
iq = ip /*Se copia el valor almacenado en ip a iq,  
el cual es la dirección de x en este momento*/
```



# PUNTEROS Y VARIABLES SIMPLES

## Paso por valor y paso por referencia

Pregunta 1: ¿Qué valor se imprime en pantalla en cada uno de estos programas?

```
void sub(int);  
main(){  
    int a=3;  
    sub(a);  
    printf("%d\n",a);  
}  
void sub(int a){  
    a=10;  
}
```

```
void sub(int *);  
main(){  
    int a=3;  
    sub(&a);  
    printf("%d\n",a);  
}  
void sub(int *p){  
    *p=10;  
}
```





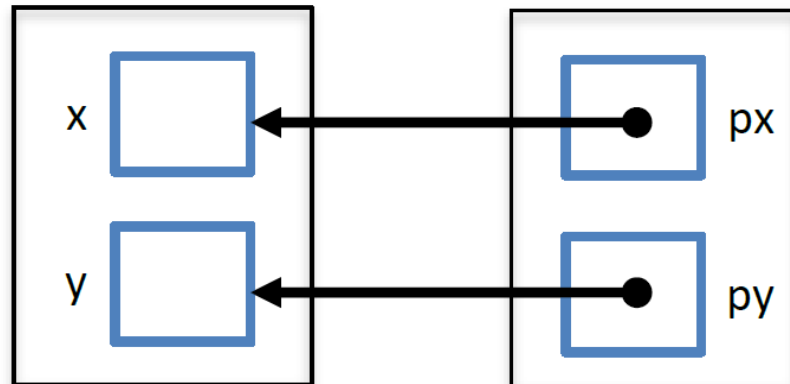
# PUNTEROS Y VARIABLES SIMPLES

Pregunta 2: ¿Son formas equivalentes de definir la función swap?

```
void swap(int x, int y){  
    int temp;  
    temp = x;  
    x = y;  
    y = temp;  
}
```

```
void swap(int *px, int  
*py){  
    int temp;  
    temp = *px;  
    *px = *py;  
    *py = temp;  
}
```

En la función  
que llama



En la función  
swap



# GRACIAS POR LA ATENCIÓN

