LECCIÓN 3: Punteros

¿Todos los lenguajes de programación permiten el uso de punteros?	No
¿Por qué se usan los punteros?	Porque optimizan el código a ejecutar
¿Qué es un puntero?	Una variable que contiene una dirección de memoria.
Forma general de la declaración de un puntero	`tipo * var_nombre;`
Declare una variable de tipo puntero a carácter denominada car	`char *car;`
¿Qué es NULL?	Es el valor nulo de un puntero
¿Cuáles son los operadores de acceso a punteros?	El de dirección (&) y el de contenido(*)
¿Qué función realiza el operador de dirección (&)?	Devuelve la dirección de memoria en la que se encuentra almacenada la variable
¿Qué función realiza el operador de contenido (*)?	Devuelve el valor contenido en la dirección de memoria almacenada en el puntero.

Dado el siguiente código:
`float n = 3.6;`
`float *ptr = &n;`
`n = 2.8;`
`printf("El valor de ptr es %f\n", *ptr);`
¿Qué valor de ptr se imprime por pantalla?
	2.8

Dado el siguiente código:
`int a, b, c, *p1, *p2;`
`p1 = &a;`
`*p1 = 1;`
`p2 = &b;`
`*p2 = 2;`
`p1 = p2;`
`*p1 = 0;`
`p2 = &c;`
`*p2 = 3;`
`printf("%d %d %d\n", a, b, c);`
¿Qué se imprime?
	1 0 3

¿Qué operaciones aritméticas se pueden realizar con punteros?	Suma, resta, asignación y comparación
¿Qué significa sumar *x* posiciones a un	Desplazarnos *x* posiciones hacia

puntero?	delante en memoria

Dado el siguiente código:
`int *v;`
`v++;`
`*v = 3;`
¿Qué posición ocupa el valor 3 en v?
	La segunda (v[1]).

¿Qué significa restar *x* posiciones a un puntero?	Desplazarnos *x* posiciones hacia detrás en memoria

Dado el siguiente código:
`int *v;`
`*v = 3;`
`*(v+2) = 2;`
`v--;`
`*v = 1;`
¿Cuáles son los valores de v?
	v = {3, 1, 2};

Cuando se comparan dos punteros, ¿qué estamos comparando realmente?	Las direcciones de memoria a las que apuntan

Dado el siguiente código:
`int main()`
`{`
`int uno, dos, *p_uno, *p_dos;`
 	
`p_uno = &uno;`
`p_dos = &dos;`
`if(p_uno == p_dos)`
`printf("Apuntan a la misma dirección\n");`
`else`
`printf("NO apuntan a la misma dirección\n");`
`system("PAUSE");`
`}`
¿Qué se imprimiría?
	NO apuntan a la misma dirección

¿Cuál es el operador de asignación para los punteros?	El =.
¿Cuáles son los ámbitos principales de las variables en C?	Locales y globales
Si se declara una variable local en la función main(), ¿se puede utilizar dentro de cualquier otra función dentro del programa?	No.
¿Cuándo acaba el tiempo de vida de una variable local?	Cuando finaliza la función en la que se ha declarado.
¿Cuándo acaba el tiempo de vida de una variable global?	Cuando finaliza el programa
¿Son iguales el paso por valor y por referencia?	No
¿Cuál es la diferencia entre paso por valor y por referencia?	En el primero se pasa una copia del valor origen y en el segundo un puntero al valor origen.
¿Cuál de los pasos de parámetros puede modificar el valor origen?	El paso por referencia

Dada la siguiente función:
`int Suma(int a, int b)`
`{`
`int resultado;`
`resultado = a + b;`
`return resultado;`
`}`
¿Cómo es el paso de parámetros?
	Por valor

¿Cuáles son las funciones de C para reserva de memoria dinámica?	malloc(), calloc(), realloc()
¿Para qué se usa free()?	Para liberar la memoria dinámica reservada.
¿Cuál es el prototipo de la función malloc()?	`void *malloc(size_t size);`
¿Qué hace calloc()?	Similar a malloc() pero inicializa a cero los valores del bloque de memoria
¿Qué hace realloc()?	Amplía el bloque de memoria reservado
Reserve memoria dinámica para un vector de 15 enteros inicializándolos a 0.	`int *v;` `v = (int*)calloc(15, sizeof(int));`

¿Es correcto este fragmento de código?¿Por qué?
`char *c, *tmp;`
`c = (char*)malloc(5 *sizeof(char));`
`tmp = (char*)realloc(c,10*sizeof(char));`
	Sí, porque amplía el bloque de memoria de *c* a partir de otro puntero *tmp*