

Cuestiones:

1. Un *array* es un tipo complejo de datos donde cada elemento ¿es del mismo tipo?
2. ¿Qué tipo de datos puede contener un *array*?
3. Si definimos un *array* de 133 elementos podemos acceder al [133]?
4. ¿Dónde empieza un *array* y de qué tipo es el índice?
5. El nombre de un *array* ¿es un puntero constante?
6. ¿Se pueden inicializar todos los *arrays*?
7. ¿Qué valor por defecto toma un *array*?
8. Si un entero ocupa 2 bytes en un sistema y declaramos un *array* de enteros, ¿Cuál es el resultado de sumar al *array* una unidad?
9. ¿Cómo se pasan los *arrays* cuando son argumentos de funciones? ¿Es necesario poner la longitud del mismo?
10. ¿Cómo se almacenan los *arrays* multidimensionales? ¿Se pueden inicializar?
11. Si tengo un *array* de 4x3 y accedo al elemento 3,2, ¿Qué número tendría que sumar a la dirección del *array* para colocarme en esa dirección?
12. Una tira es un *array* de caracteres, pero ¿Con qué peculiaridad?
13. ¿Cómo se pueden inicializar las tiras?
14. Cuando se lee un *array* con *scanf* ¿es necesario utilizar el operador "&"?
15. Si hago: `static char *array[5]` ¿estoy haciendo un *array* de 5 punteros o un puntero a un *array* de cinco elementos?
16. ¿Es usual utilizar *scanf* para leer tiras?
17. ¿De qué tipo es *gets*?
18. Si se ocurre un error en *gets* ¿qué devuelve la función?
19. La función *puts* ¿en qué convierte el carácter nulo de la tira?
20. Un *struct* es un tipo complejo de datos donde cada elemento ¿es del mismo tipo?
21. ¿Qué es un *tag* o patrón?
22. ¿Se pueden inicializar estructuras? ¿en qué condiciones?
23. ¿Con qué operador se accede a un miembro de una estructura?
24. ¿Qué diferencia hay entre "." y "->"?
25. Si en una *union* declaramos un *array* de 100 enteros de dos bytes y otro de 50 flotantes de 4 bytes, ¿Cuánto ocupará la *union*?