

# 14 Lecture - CS201

## Important Subjective

- 1. What is a pointer in C programming?**  
A pointer is a variable that stores a memory address. It allows programmers to directly manipulate memory and is useful for efficient dynamic memory allocation.
- 2. How do you declare a pointer variable in C?**  
A pointer variable is declared by adding an asterisk (\*) before the variable name, for example:  
`int *ptr;`
- 3. What is the purpose of the ampersand (&) operator in C?**  
The ampersand (&) operator is used to get the address of a variable in memory, for example: #
- 4. How do you access the value pointed to by a pointer in C?**  
You can access the value pointed to by a pointer by using the dereference operator (\*) before the pointer variable, for example: `*ptr;`
- 5. What is a null pointer in C?**  
A null pointer is a pointer that does not point to any valid memory location. It is represented in C by the value 0 or NULL.
- 6. How do you use pointers to dynamically allocate memory in C?**  
You can use the `malloc()` function to dynamically allocate memory in C, and then use a pointer to access the allocated memory, for example: `int ptr = (int) malloc(sizeof(int));`
- 7. How do you pass pointers as function arguments in C?**  
You can pass pointers as function arguments by declaring the function parameter as a pointer and then passing the memory address of the variable as an argument, for example: `void myFunction(int *ptr);`
- 8. What is a pointer arithmetic in C?**  
Pointer arithmetic in C involves manipulating the memory address stored in a pointer variable using arithmetic operations, such as addition or subtraction.
- 9. What is a void pointer in C?**  
A void pointer is a special type of pointer that can point to any type of data. It is useful for generic programming and dynamic memory allocation.
- 10. How do you use pointers to manipulate arrays in C?**  
You can use pointers to manipulate arrays in C by using pointer arithmetic to access array elements, for example: `int arr[5] = {1, 2, 3, 4, 5}; int *ptr = arr; printf("%d", *(ptr + 2)); // prints 3`