

## **1.What is a pointer?**

Answer:

A pointer is a variable that stores the memory address of another variable.

Pointers are mainly used for:

Dynamic memory allocation

Passing data efficiently to functions

Working with arrays and strings

Accessing memory directly

Pointers improve program performance and allow better memory management.

## **2.What is the difference between array and pointer?**

Answer:

An array is a collection of elements stored in contiguous memory locations, whereas a pointer stores the address of a variable.

An array name always points to the same memory location, but a pointer can be reassigned to point to different locations.

Arrays have fixed size, while pointers provide flexibility.

## **3. What is the difference between malloc() and calloc()?**

Answer:

Both malloc() and calloc() are used for dynamic memory allocation.

malloc() allocates memory but does not initialize it, so it contains garbage values.

calloc() allocates memory and initializes all bytes to zero.

calloc() is safer when initialization is required.

## **4. What is the difference between ++i and i++?**

Answer:

++i is pre-increment, where the value is incremented first and then used.

i++ is post-increment, where the value is used first and then incremented.

This difference is important in expressions and loops.

## **5. What is static keyword in C?**

Answer:

The static keyword is used to retain the value of a variable between function calls.

A static variable:

Is initialized only once

Exists throughout the program execution

Is stored in the data segment, not the stack

It is also used to restrict global variables and functions to a single file.

## **6. What is const keyword?**

Answer:

The const keyword is used to make a variable read-only, meaning its value cannot be changed after initialization.

It helps in:

Protecting data

Improving code readability

Avoiding accidental modifications

## **7. What is recursion?**

Answer:

Recursion is a technique where a function calls itself to solve a problem.

A recursive function must have:

A base condition to stop recursion

A recursive call that reduces the problem size

Recursion is commonly used in problems like factorial, Fibonacci, and tree traversal.

### **8.What is a segmentation fault?**

Answer:

A segmentation fault occurs when a program tries to access invalid or unauthorized memory.

Common causes include:

Dereferencing uninitialized or NULL pointers

Accessing array elements out of bounds

Improper memory handling

It results in abnormal program termination.

---

### **9. Difference between struct and union**

Answer:

In a struct, each member gets separate memory, so all members can be used at the same time.

In a union, all members share the same memory location, so only one member can be used at a time.

Structures consume more memory, while unions save memory.

---

### **10. Difference between break and exit()**

Answer:

break is used to exit from a loop or switch statement, while exit() is used to terminate the entire program.

exit() is defined in stdlib.h.

## Storage Classes in C

Storage classes in C define four important things about a variable:

1. Scope – where the variable can be accessed
2. Lifetime – how long the variable exists in memory
3. Storage location – where the variable is stored
4. Default initial value

C provides four storage classes:

auto

register

static

extern

### 1. auto Storage Class

auto is the default storage class for all local variables inside a function.

If we do not mention any storage class, the variable is automatically treated as auto.

#### Key Points:

**Scope:** Inside the block or function

**Lifetime:** Exists only during function execution

**Memory:** Stack memory

**Default value:** Garbage value

#### Example:

```
void fun() {  
    auto int x = 10;  
}
```

### 2.register Storage Class

:

register suggests the compiler to store the variable in a CPU register instead of RAM for faster access.

#### Key Points:

**Scope:** Local to block

**Lifetime:** Till the block execution

**Memory:** CPU register (if available)

We cannot use & (address) operator

**Example:**

register int count;

**3. static Storage Class**

A static variable retains its value even after the function call ends. It is initialized only once and stored in the data segment.

**a) Static Local Variable**

**Scope:** Inside the function

**Lifetime:** Entire program execution

**Example:**

```
void fun() {  
    static int x = 0;  
    x++;  
    printf("%d", x);  
}
```

Each function call prints incremented value.

**b) Static Global Variable**

**Scope:** Only within the same file

Prevents access from other files

**4. extern Storage Class**

extern is used to access a global variable declared in another file. It does not allocate memory, it only refers to an existing variable.

**Key Points:**

**Scope:** Global

**Lifetime:** Entire program execution

Used in multi-file programs

**Example:**

extern int total;