18-12-2025

A. Question And Answers

1. **What is C language and why is it called a middle-level language?**

   C is a procedural, general-purpose programming language developed by Dennis Ritchie. It is called a **middle-level language** because it combines **low-level features** like pointers and direct memory access with **high-level features** such as functions, loops, and structured programming—making it powerful for both system-level and application development.

2. **What is a pointer? Why is it used?**

   A pointer is a variable that stores the memory address of another variable. It is used to enable **direct memory access**, **efficient data handling**, **dynamic memory allocation**, and **call by reference**, which helps in writing faster and memory-efficient programs.

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

Malloc()	Calloc()
Allocates single block	Allocates multiple blocks
Does not initialize memory	Initializes memory to 0
Faster	Slightly slower

4. **What is the difference between structure and union?**

Structure	Union
Each member has its own memory	All members share same memory
More memory usage	Less memory usage
All members accessible	One member at a time

5. **What is segmentation fault?**

   A segmentation fault is a runtime error that occurs when a program tries to access memory it is not allowed to access. It commonly happens due to **invalid pointers, accessing array out of bounds, or using freed/uninitialized memory**, and it usually causes the program to crash immediately.

6. **What is the difference between call by value and call by reference?**

- **Call by value** passes a **copy of the variable**, so changes made inside the function do **not affect the original value**.
- **Call by reference** passes the **address of the variable** (using pointers), so changes made inside the function **directly affect the original value**.

7. **What is an array? How is it stored in memory?**

   An array is a collection of elements of the same data type stored under a single name. In memory, an array is stored in **contiguous memory locations**, which allows **fast access** to elements using an index. Each element occupies equal-sized memory, and its address is calculated using the base address plus the index offset.

8. **What is the difference between array and pointer?**

   An array is a fixed-size collection of elements stored in contiguous memory, while a pointer is a variable that stores the address of a memory location.
   An array's name represents the base address and **cannot be reassigned**, whereas a pointer **can be reassigned to point to different locations**. Arrays allocate memory at declaration, while pointers can be used for **dynamic memory access**.

9. **What are storage classes in C?**

   Storage classes in C define the scope, lifetime, and visibility of variables in a program. They control where a variable is stored, how long it exists, and who can access it. The main storage classes are **auto, register, static, and extern**, each used based on memory and access requirements.

10. **What is a dangling pointer?**

    A dangling pointer is a pointer that refers to a memory location which has already been freed or no longer exists.
    It occurs when memory is deallocated but the pointer is not reset, leading to **undefined behavior or crashes** if accessed.

B. Explain about storage classes.

Storage Classes in C Programming

Definition

Storage classes in C define **four important properties of a variable**:

- **Scope –** where the variable can be accessed
- **Lifetime** – how long the variable exists in memory
- **Visibility** – which parts of the program can see it
- **Default value** – initial value if not assigned

Types of Storage Classes in C

C supports four storage classes:

1. auto
2. register
3. static
4. extern

1. auto Storage Class

➢ Default storage class for **local variables**
➢ Stored in **stack memory**
➢ Scope: Inside the block only
➢ Default value: **Garbage**

Example:

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

❖ Note: Writing auto is optional

2. register Storage Class

➢ Suggests storing variable in CPU register
➢ Faster access than memory
➢ Address (&) cannot be used

Example:

```
register int count;
```

❖ Used in loops and counters

3. static Storage Class

➢ Retains value even after function execution
➢ Initialized only once
➢ Stored in data segment
➢ Default value: 0

Example:

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

Output: 1 2 3 (on repeated calls)

- ❖ Used when data persistence is needed

4. extern Storage Class

➢ Used to access global variables defined in another file
➢ Does not allocate memory (only declaration)

Example:

```
extern int a;
```

- ❖ Used in multi-file programs