**TOP 100 C LANGUAGE INTERVIEW QUESTIONS**

**Basic C Language Questions**

1. **What is C language?**  
   *Answer:* C is a general-purpose, procedural programming language developed by Dennis Ritchie.
2. **Why is C called a middle-level language?**  
   *Answer:* C combines low-level hardware access with high-level features, making it a middle-level language.
3. **What are the key features of C?**  
   *Answer:* Simple syntax, portability, structured programming, pointers, and memory management.
4. **Explain the structure of a C program.**  
   *Answer:* A C program typically includes headers, a main() function, and other functions.
5. **What are header files?**  
   *Answer:* Header files contain declarations of functions and macros. Example: #include <stdio.h>.

**Data Types and Variables**

1. **What are the basic data types in C?**  
   *Answer:* int, float, double, char.
2. **What is a variable?**  
   *Answer:* A variable is a storage location with a name and a type to store data.
3. **What is the difference between int and float?**  
   *Answer:* int stores whole numbers, while float stores decimal numbers.
4. **What is a constant in C?**  
   *Answer:* A constant is a variable whose value cannot be changed. Declared using const.
5. **What is the use of the sizeof() operator?**  
   *Answer:* It returns the size of a data type or variable in bytes.

**Operators and Expressions**

1. **What are the different types of operators in C?**  
   *Answer:* Arithmetic, relational, logical, bitwise, assignment, and conditional operators.
2. **What is the difference between == and =?**  
   *Answer:* == checks equality, while = assigns a value.
3. **What is a ternary operator?**  
   *Answer:* It is a shorthand for an if-else condition: condition ? true\_value : false\_value.
4. **Explain precedence and associativity of operators.**  
   *Answer:* Precedence determines the order of evaluation. Associativity decides the direction (left-to-right or right-to-left).
5. **What is the modulus operator %?**  
   *Answer:* It returns the remainder of division between two integers.

**Control Statements**

1. **What are control statements in C?**  
   *Answer:* if, else, switch, while, do-while, for.
2. **What is the difference between while and do-while loops?**  
   *Answer:* while checks the condition before execution; do-while checks after.
3. **Explain the break and continue statements.**  
   *Answer:* break exits a loop, while continue skips the current iteration.
4. **How does a switch statement work?**  
   *Answer:* It matches the value of an expression with case labels and executes the corresponding block.
5. **What is a goto statement?**  
   *Answer:* It provides an unconditional jump to a labeled statement, but it’s generally discouraged.

**Functions**

1. **What is a function in C?**  
   *Answer:* A function is a block of code that performs a specific task and can be reused.
2. **What is the difference between call by value and call by reference?**  
   *Answer:* Call by value passes a copy of the argument, while call by reference passes the address.
3. **What is the use of the return statement?**  
   *Answer:* It exits a function and optionally returns a value to the caller.
4. **Can we have a function inside a function in C?**  
   *Answer:* No, C does not allow nested functions.
5. **What is recursion?**  
   *Answer:* Recursion is a function calling itself.

**Arrays and Strings**

1. **What is an array?**  
   *Answer:* An array is a collection of elements of the same type stored in contiguous memory.
2. **How to declare and initialize an array?**  
   *Answer:* int arr[5] = {1, 2, 3, 4, 5};
3. **What is the difference between a string and a character array?**  
   *Answer:* A string ends with a null character \0, while a character array may not.
4. **How to read a string in C?**  
   *Answer:* Using scanf() or gets().
5. **What is a multi-dimensional array?**  
   *Answer:* An array with more than one dimension, like a matrix: int arr[3][3];.

**Pointers**

1. **What is a pointer in C?**  
   *Answer:* A pointer is a variable that stores the address of another variable.
2. **How to declare a pointer?**  
   *Answer:* int \*p;
3. **What is the difference between \* and &?**  
   *Answer:* \* is the dereference operator, & is the address-of operator.
4. **What is a NULL pointer?**  
   *Answer:* A pointer that does not point to any valid memory location.
5. **What is pointer arithmetic?**  
   *Answer:* Performing operations like addition or subtraction on pointers.

**Structures and Unions**

1. **What is a structure in C?**  
   *Answer:* A structure is a user-defined data type that groups different data types.

struct Student {

int id;

char name[20];

};

1. **How to access structure members?**  
   *Answer:* Using the dot operator (.) for variables and arrow operator (->) for pointers.  
   Example: student1.name or ptr->id.
2. **What is the difference between structure and union?**  
   *Answer:* Structures allocate separate memory for each member, while unions share memory.
3. **How to declare and initialize a union?**  
   *Answer:*

union Data {

int i;

float f;

};

union Data d = {10};

1. **What is a typedef?**  
   *Answer:* typedef creates a new name for an existing data type.  
   Example: typedef unsigned int uint;

**Dynamic Memory Allocation**

1. **What are dynamic memory allocation functions in C?**  
   *Answer:* malloc(), calloc(), realloc(), and free().
2. **What is the difference between malloc() and calloc()?**  
   *Answer:* malloc() allocates memory without initializing, while calloc() initializes with zero.
3. **What is the purpose of free()?**  
   *Answer:* It deallocates memory allocated dynamically.
4. **What is the syntax of malloc()?**  
   *Answer:*

int \*ptr = (int\*)malloc(sizeof(int));

1. **What happens if free() is not used?**  
   *Answer:* It leads to memory leaks, where allocated memory is not released.

**File Handling**

1. **What is file handling in C?**  
   *Answer:* It allows reading and writing files using functions like fopen(), fread(), and fwrite().
2. **What are the different modes to open a file?**  
   *Answer:* r, w, a, r+, w+, a+.
3. **What is the syntax for opening a file?**  
   *Answer:*

FILE \*fp = fopen("file.txt", "r");

1. **How to close a file?**  
   *Answer:* Using fclose(fp);.
2. **What is the difference between fgets() and gets()?**  
   *Answer:* fgets() reads a line with a limit, while gets() reads until a newline but is unsafe.

**Preprocessors and Macros**

1. **What is a preprocessor in C?**  
   *Answer:* Preprocessors process code before compilation. Example: #include, #define.
2. **What is the use of #define?**  
   *Answer:* It defines constants or macros.  
   Example: #define PI 3.14.
3. **What is the difference between #include <file> and #include "file"?**  
   *Answer:* Angle brackets search in system directories, double quotes search in the current directory.
4. **What is a macro?**  
   *Answer:* A macro is a fragment of code given a name and expanded by the preprocessor.
5. **What is the use of conditional compilation?**  
   *Answer:* It allows compiling certain parts of the program based on conditions using directives like #ifdef.

**Strings and Character Handling**

1. **What is a string in C?**  
   *Answer:* A string is a sequence of characters ending with a null character (\0).
2. **How to compare two strings?**  
   *Answer:* Using strcmp() function.
3. **How to find the length of a string?**  
   *Answer:* Using strlen() function.
4. **What is the use of strcpy()?**  
   *Answer:* It copies one string into another.
5. **How to concatenate two strings?**  
   *Answer:* Using strcat() function.

**Bitwise Operators**

1. **What are bitwise operators in C?**  
   *Answer:* Operators like &, |, ^, ~, <<, >> used to manipulate bits.
2. **What is the difference between & and &&?**  
   *Answer:* & is a bitwise AND operator, && is a logical AND operator.
3. **How to check if a number is even or odd using bitwise operators?**  
   *Answer:*

if (num & 1) printf("Odd");

else printf("Even");

1. **What is a bitwise shift operator?**  
   *Answer:* << shifts bits left, and >> shifts bits right.
2. **How to swap two numbers without a temporary variable?**  
   *Answer:*

a = a ^ b;

b = a ^ b;

a = a ^ b;

**Memory Management**

1. **What is the stack and heap in C?**  
   *Answer:* The stack is for static memory allocation, and the heap is for dynamic allocation.
2. **What causes a stack overflow?**  
   *Answer:* When the program uses more stack space than available, often due to deep recursion.
3. **What is a memory leak?**  
   *Answer:* It occurs when dynamically allocated memory is not freed.
4. **How to avoid memory leaks?**  
   *Answer:* By using free() after malloc() or calloc().
5. **What is a dangling pointer?**  
   *Answer:* A pointer pointing to memory that has been freed.

**Advanced Concepts**

1. **What is a static variable in C?**  
   *Answer:* A static variable retains its value across function calls and is initialized only once.
2. **What is the difference between static and global variables?**  
   *Answer:* Static variables have a local scope but retain their value, while global variables are accessible throughout the program.
3. **What is an extern variable?**  
   *Answer:* An extern variable is declared in one file and defined in another, making it accessible across files.
4. **What are storage classes in C?**  
   *Answer:* Storage classes define variable scope, lifetime, and linkage: auto, static, extern, and register.
5. **What is a volatile keyword?**  
   *Answer:* volatile tells the compiler not to optimize the variable as its value may change unexpectedly (e.g., hardware registers).

**Debugging and Error Handling**

1. **What is a segmentation fault?**  
   *Answer:* It occurs when a program tries to access restricted memory.
2. **What is a bus error?**  
   *Answer:* It occurs when the CPU attempts to access unaligned memory.
3. **How do you debug a C program?**  
   *Answer:* Using tools like gdb (GNU Debugger), print statements, and analyzing core dumps.
4. **What is the purpose of assert()?**  
   *Answer:* It helps in debugging by terminating the program if a condition is false.
5. **What are some common runtime errors in C?**  
   *Answer:* Segmentation faults, buffer overflows, memory leaks, and division by zero.

**Best Practices in C Programming**

1. **Why should you avoid using gets()?**  
   *Answer:* It does not check buffer limits and can lead to buffer overflow. Use fgets() instead.
2. **How do you ensure code readability?**  
   *Answer:* By using meaningful variable names, proper indentation, and comments.
3. **Why is it important to free memory in C?**  
   *Answer:* To avoid memory leaks and ensure efficient memory use.
4. **What is the importance of header guards?**  
   *Answer:* They prevent multiple inclusions of the same header file using #ifndef and #define.
5. **What is the purpose of const keyword?**  
   *Answer:* It declares variables whose values cannot be changed after initialization.

**Multithreading and Concurrency**

1. **Does C support multithreading?**  
   *Answer:* C itself does not have built-in support, but multithreading can be implemented using libraries like POSIX threads (pthread).
2. **What are race conditions?**  
   *Answer:* They occur when two or more threads access shared data simultaneously, leading to unpredictable results.
3. **What is a mutex?**  
   *Answer:* A mutex (mutual exclusion) is used to protect shared resources in multithreading.
4. **How to create a thread in C?**  
   *Answer:* Using the pthread\_create() function from the pthread library.
5. **What is thread synchronization?**  
   *Answer:* It ensures that threads execute in a predictable order when accessing shared resources.

**Inline Functions and Macros**

1. **What is an inline function?**  
   *Answer:* A function that is expanded in place of a call, reducing function call overhead.
2. **What is the difference between macros and inline functions?**  
   *Answer:* Macros are preprocessed, while inline functions are compiled and type-checked.
3. **How to define an inline function?**  
   *Answer:* Using the inline keyword:

inline int square(int x) { return x \* x; }

1. **What are the disadvantages of macros?**  
   *Answer:* No type checking, difficult debugging, and increased code size.
2. **What is the advantage of using inline functions over macros?**  
   *Answer:* Type safety, easier debugging, and better error handling.

**Miscellaneous Questions**

1. **What is the difference between C and C++?**  
   *Answer:* C is procedural, while C++ supports object-oriented programming with classes and inheritance.
2. **What are dangling pointers?**  
   *Answer:* Pointers that reference memory that has been freed.
3. **What is the role of main() in a C program?**  
   *Answer:* It is the entry point of the program, where execution starts.
4. **What is the output of printf("%d", sizeof("Hello"));?**  
   *Answer:* It prints 6 because the string includes the null terminator (\0).
5. **Why is C still widely used?**  
   *Answer:* Because of its efficiency, control over hardware, and use in system-level programming, embedded systems, and operating systems.