**SAMPLE QUESTIONS PREPARED BY ROHIT:-**

**C programming**

**1. What is difference between i++ and ++i?**

1) The expression 'i++' returns the old value and then increments i. The expression ++i increments the value and returns new value.
2) Precedence of postfix ++ is higher than that of prefix ++.
3) Associativity of postfix ++ is left to right and associativity of prefix ++ is right to left.
4) In C++, ++i can be used as l-value, but i++ cannot be. In C, they both cannot be used as l-value.

**2. What Does Static Variable Mean?**

Static variable is available to a C application, throughout the life time. At the time of starting the program execution, static variables allocations takes place first. In a scenario where one variable is to be used by all the functions (which is accessed by main () function), then the variable need to be declared as static in a C program.

**3. What Is A Null Pointer?**

A null pointer is a special pointer value that is known not to point anywhere. It means that no other valid pointer, to any other variable or array cell or anything else, will ever compare equal to a null pointer.

**4. What Is The Difference Between Call By Value And Call By Reference?**

When using Call by Value, you are sending the value of a variable as parameter to a function, whereas Call by Reference sends the address of the variable. Also, under Call by Value, the value in the parameter is not affected by whatever operation that takes place, while in the case of Call by Reference, values can be affected by the process within the function.

**5. What Are Header Files And What Are Its Uses In C Programming?**

Header files are also known as library files. They contain two essential things: the definitions and prototypes of functions being used in a program. Simply put, commands that you use in C programming are actually functions that are defined from within each header files. Each header file contains a set of functions. For example: stdio.h is a header file that contains definition and prototypes of commands like printf and scanf.

**6. Differentiate Source Codes from Object Codes**

Source codes are codes that were written by the programmer. It is made up of the commands and other English-like keywords that are supposed to instruct the computer what to do. However, computers would not be able to understand source codes. Therefore, source codes are compiled using a compiler. The resulting outputs are object codes, which are in a format that can be understood by the computer processor. In C programming, source codes are saved with the file extension .C, while

object codes are saved with the file extension .OBJ

**7. In C programming, how do you insert quote characters (' and ") into the output screen?**

This is a common problem for beginners because quotes are normally part of a printf statement. To insert the quote character as part of the output, use the format specifiers \' (for single quote), and \" (for double quote).

**8. Why is it that not all header files are declared in every C program?**

The choice of declaring a header file at the top of each C program would depend on what commands/functions you will be using in that program. Since each header file contains different function definitions and prototype, you would be using only those header files that would contain the functions you will need. Declaring all header files in every program would only increase the overall file size and load of the program, and is not considered a good programming style.

**9. When is the "void" keyword used in a function?**

When declaring functions, you will decide whether that function would be returning a value or not. If that function will not return a value, such as when the purpose of a function is to display some outputs on the screen, then "void" is to be placed at the leftmost part of the function header. When a return value is expected after the function execution, the data type of the return value is placed instead of "void".

**10. What is wrong in this statement? scanf("%d",whatnumber);**

An ampersand & symbol must be placed before the variable name whatnumber. Placing & means whatever integer value is entered by the user is stored at the "address" of the variable name. This is a common mistake for programmers, often leading to logical errors.

**11. What are preprocessor directives?**

Preprocessor directives are placed at the beginning of every C program. This is where library files are specified, which would depend on what functions are to be used in the program. Another use of preprocessor directives is the declaration of constants.Preprocessor directives begin with the # symbol.

**12. Describe the order of precedence with regards to operators in C.**

Order of precedence determines which operation must first take place in an operation statement or conditional statement. On the top most level of precedence are the unary operators !, +, - and &. It is followed by the regular mathematical operators (*, / and modulus % first, followed by + and -). Next in line are the relational operators <, <=, >= and >. This is then followed by the two equality operators == and !=. The logical operators && and || are next evaluated. On the last level is the assignment operator =.

**13. Why is C language being considered a middle level language?**

This is because C language is rich in features that make it behave like a high level language while at the same time can interact with hardware using low level methods. The use of a well structured approach to programming, coupled with English-like words used in functions, makes it act as a high level language. On the other hand, C can directly access memory structures similar to assembly language routines.

**14. What are structure types in C?**

Structure types are primarily used to store records. A record is made up of related fields. This makes it easier to organize a group of related data.

**15. What are pointers?**

Pointers point to specific areas in the memory. Pointers contain the address of a variable, which in turn may contain a value or even an address to another memory.

**16. Can you pass an entire structure to functions?**

Yes, it is possible to pass an entire structure to a function in a call by method style. However, some programmers prefer declaring the structure globally, then pass a variable of that structure type to a function. This method helps maintain consistency and uniformity in terms of argument type.

**17. Are comments included during the compilation stage and placed in the EXE file as well?**

No, comments that were encountered by the compiler are disregarded. Comments are mostly for the guidance of the programmer only and do not have any other significant use in the program functionality.

**SAMPLE QUESTIONS PREPARED BY ASHOK:-**

## 1. What is Singly Linked Lists

In a *singly linked* list, each node stores a reference to an object that is an element of the sequence, as well as a reference to the next node of the list. It does not store any pointer or reference to the previous node. To store a single linked list, only the reference or pointer to the first node in that list must be stored. The last node in a single linked list points to nothing.

## 2. Explain Doubly Linked Lists

A doubly linked list is a data structure that consists of a set of sequentially **l**inked records called nodes. Each node contains three fields: two link fields (references to the previous and to the next node in the sequence of nodes) and one data field.

## 3. List the basic operations carried out in a linked list?

The basic operations carried out in a linked list include:

- Creation of a list
- Insertion of a node
- Deletion of a node
- Modification of a node
- Traversal of the list

## 4. Explain Circular Linked List?

Circular Linked List is a variation of Linked list in which the first element points to the last element and the last element points to the first element. Both Singly Linked List and Doubly Linked List can be made into a circular linked list.

**5. State the difference between arrays and linked lists?**

| Arrays | Linked Lists |
|---|---|
| Size of an array is fixed | Size of a list is variable |
| It is necessary to specify the number of elements during declaration. | It is not necessary to specify the number of elements during declaration |
| Insertions and deletions are somewhat difficult | Insertions and deletions are carried out easily |
| It occupies less memory than a linked list for the same number of elements | It occupies more memory |

**6. List out the disadvantages of using a linked list**

- Searching a particular element in a list is difficult and time consuming
- A linked list will use more storage space than an array to store the same number of elements

**7. Define Binary Tree?**

A binary tree is a finite set of nodes which is either empty or consists of a root and two disjoint binary trees called the left sub-tree and right sub-tree.

**8. Define Degree of a Node**

The total number of sub-trees attached to that node is called the degree of the node. If the degree is zero, it is called a terminal or leaf node of a tree.

**9. Define depth and height of a node**

For any node ni, the depth of ni is the length of the unique path from the root to ni. The height of ni is the length of the longest path from ni to a leaf.

**10. Define a full binary tree**

A full binary tree is a tree in which all the leaves are on the same level and every non-leaf node has exactly two children. A full binary tree is a tree in which every node other than the leaves has two children.

**11. What are the tasks performed during inorder traversal?**

- Traverse the left sub-tree
- Process the root node
- Traverse the right sub-tree

**12. What are the tasks performed during postorder traversal?**

- Traverse the left sub-tree
- Traverse the right sub-tree
- Process the root node

**13. What are the tasks performed during preorder traversal?**

- Process the root
- Traverse the left sub-tree
- Traverse the right sub-tree

**14. Define a binary search tree?**

A binary search tree is a special binary tree, which is either empty or it should satisfy the following characteristics:

Every node has a value and no two nodes should have the same value (i.e) the values in the binary search tree are distinct

- The values in any left sub-tree is less than the value of its parent node
- The values in any right sub-tree is greater than the value of its parent node
- The left and right sub-trees of each node are again binary search trees

**15. Define AVL Trees?**

AVL tree is a self-balancing Binary Search Tree (BST) where the difference between heights of left and right subtrees cannot be more than one for all nodes.

**16. What do you mean by balance factor of a node in AVL tree?**

The height of left subtree minus height of right subtree is called balance factor of a node in AVL tree. The balance factor may be either 0 or +1 or -1. The height of an empty tree is -1