Q1. Why we need to do algorithm analysis?

A problem can be solved in more than one ways. So, many solution algorithms can be derived for a given problem. We analyze available algorithms to find and implement the best suitable algorithm.

Q2. What are the criteria of algorithm analysis?

An algorithm are generally analyzed on two factors − time and space. That is, how much **execution** time and how much **extra space** required by the algorithm.

Q3. What is asymptotic analysis of an algorithm?

Asymptotic analysis of an algorithm, refers to defining the mathematical boundation/framing of its run-time performance. Using asymptotic analysis, we can very well conclude the best case, average case and worst case scenario of an algorithm.

Q4. What are asymptotic notations?

Asymptotic analysis can provide three levels of mathematical binding of execution time of an algorithm −

* Best case is represented by Ω(n) notation.
* Worst case is represented by Ο(n) notation.
* Average case is represented by Θ(n) notation.

Q5. What is linear data structure?

A linear data-structure has sequentially arranged data items. The next time can be located in the next memory address. It is stored and accessed in a sequential manner. Array and list are example of linear data structure.

Q6.What are common operations that can be performed on a data-structure?

The following operations are commonly performed on any data-structure −

* **Insertion** − adding a data item
* **Deletion** − removing a data item
* **Traversal** − accessing and/or printing all data items
* **Searching** − finding a particular data item
* **Sorting** − arranging data items in a pre-defined sequence

Q7 Briefly explain the approaches to develop algorithms.

There are three commonly used approaches to develop algorithms −

* **Greedy Approach** − finding solution by choosing next best option
* **Divide and Conquer** − diving the problem to a minimum possible sub-problem and solving them independently
* **Dynamic Programming** − diving the problem to a minimum possible sub-problem and solving them combinedly

Q8. What are some examples of divide and conquer algorithms?

The below given problems find their solution using divide and conquer algorithm approach −

* Merge Sort
* Quick Sort
* Binary Search
* Strassen's Matrix Multiplication
* Closest pair (points)

Q9. Why do we use stacks?

Stacks follows LIFO method and addition and retrieval of a data item takes only Ο(n) time. Stacks are used where we need to access data in the reverse order or their arrival. Stacks are used commonly in recursive function calls, expression parsing, depth first traversal of graphs etc.

Q10.What operations can be performed on Queues?

The below operations can be performed on a stack −

* **enqueue()** − adds an item to rear of the queue
* **dequeue()** − removes the item from front of the queue
* **peek()** − gives value of front item without removing it
* **isempty()** − checks if stack is empty
* **isfull()** − checks if stack is full

Q11.What is linear searching?

Linear search tries to find an item in a sequentially arranged data type. These sequentially arranged data items known as array or list, are accessible in incrementing memory location. Linear search compares expected data item with each of data items in list or array. The average case time complexity of linear search is Ο(n) and worst case complexity is Ο(n2). Data in target arrays/lists need not to be sorted.

Q12. How insertion sort and selection sorts are different?

Both sorting techniques maintains two sub-lists, sorted and unsorted and both take one element at a time and places it into sorted sub-list. Insertion sort works on the current element in hand and places it in the sorted array at appropriate location maintaining the properties of insertion sort. Whereas, selection sort searches the minimum from the

Q13. How quick sort works?

Quick sort uses divide and conquer approach. It divides the list in smaller 'partitions' using 'pivot'. The values which are smaller than the pivot are arranged in the left partition and greater values are arranged in the right partition. Each partition is recursively sorted using quick sort.

Q14. What is tree traversal?

Tree traversal is a process to visit all the nodes of a tree. Because, all nodes are connected via edges (links) we always start from the root (head) node. There are three ways which we use to traverse a tree −

* In-order Traversal
* Pre-order Traversal
* Post-order Traversal

Q15. What is an AVL Tree?

AVL trees are height balancing binary search tree. AVL tree checks the height of left and right sub-trees and assures that the difference is not more than 1. This difference is called Balance Factor.

BalanceFactor = height(left-sutree) − height(right-sutree)

Q16. How many spanning trees can a graph has?

It depends on how connected the graph is. A complete undirected graph can have maximum nn-1 number of spanning trees, where n is number of nodes.

Q17. What is a heap in data structure?

Heap is a special balanced binary tree data structure where root-node key is compared with its children and arranged accordingly. A min-heap, a parent node has key value less than its childs and a max-heap parent node has value greater than its childs.

Q18. What is a recursive function?

A recursive function is one which calls itself, directly or calls a function that in turn calls it. Every recursive function follows the recursive properties − **base criteria** where functions stops calling itself and **progressive approach** where the functions tries to meet the base criteria in each iteration.

Q19. What is interpolation search technique?

Interpolation search is an improved variant of binary search. This search algorithm works on the probing position of required value.

Q20. What is the prefix and post fix notation of (a + b) \* (c + d) ?

Prefix Notation − \* + a b + c d

Postfix Notation − a b + c d + \*

Q21 Mention the features of ADT.

a. Modularity

i. Divide program into small functions

ii. Easy to debug and maintain

iii. Easy to modify

b. Reuse

i. Define some operations only once and reuse them in future

c. Easy to change the implementation

Q22 Define List ADT

A list is a sequence of zero or more elements of a given type. The list is represented as sequence of elements separated by comma.

A1, A2, A3…..AN

Where N>0 and A is of type element

Q23 What is a circular linked list?

A circular linked list is a special type of linked list that supports traversing from the end of the list to the beginning by making the last node point back to the head of the list.

Q24. What are the methods to implement stack in C?

The methods to implement stacks are:

● Array based

● Linked list based

Q25. Define double ended queue

● It is a special type of queue that allows insertion and deletion of elements at both

Q26. Write the advantages of threaded binary tree.

The difference between a binary tree and the threaded binary tree is that in the binary trees the nodes are null if there is no child associated with it and so there is no way to traverse back. But in a threaded binary tree we have threads associated with the nodes i.e they either are linked to the predecessor or successor in the in order traversal of the nodes. This helps us to traverse further or backward in the in order traversal fashion.

Q27. How heap can it be used to represent a priority queue?

A priority queue is a different kind of queue, in which the next element to be removed is defined by (possibly) some other criterion. The most common way to implement a priority queue is to use a different kind of binary tree, called a heap. A heap avoids the long paths that can arise with binary search trees.

Q28. Define biconnected graph.

A connected undirected graph is biconnected if there are no vertices whose removal disconnects the rest of the graph.

Q29. Define NP - Complete

NP is the class of decision problems for which a given proposed solution for a given input can be checked quickly to see if it is really a solution.

Q30. What is a simple graph?

A simple graph is a graph, which has not more than one edge between a pair of nodes than such a graph is called a simple graph.