CODERS LODGE

DATA STRUCTURE & ALGORITHM

VIVA QUESTIONS

Kindly read the instructions carefully

1. **All these questions are important for examination and interview point of view, so practice them well.**
2. **If you have any doubt or facing any problem regarding these questions you can mail us at coderslodgeofficial@gmail.com or drop a message in our WhatsApp or telegram group.**
3. **If you want to support us, give your valuable feedback so that next time we can improve while interacting with you.**
4. ***Reminder*-Practice all questions well it will build your concept clear and you can easily score good in your exams.**

Connect with us

- If you want to join our **WhatsApp** community then mail us at: - coderslodgeofficial@gmail.com
- Join our **Telegram group**: - https://t.me/coderslodgeofficial
- Like our **Facebook page**: - https://www.facebook.com/coderslodge
- Follow us on **Instagram**:- https://www.instagram.com/coderslodge/
- Follow us on **Twitter**: - https://twitter.com/CodersLodge
- Follow us on **LinkedIn**:- https://www.linkedin.com/company/coderslodge

1 **What is data-structure?**

Data structure is a way of defining, storing & retrieving of data in a structural & systematic way. A data structure may contain different type of data items.

2 **Differentiate between file and structure storage structure.**

The key difference between both the data structure is the memory area that is being accessed. When dealing with the structure that resides the main memory of the computer system, this is referred to as storage structure. When dealing with an auxiliary structure, we refer to it as file structures.

3 **What are various data-structures available?**

Data structure availability may vary by programming languages. Commonly available data structures are list, arrays, stack, queues, graph, tree etc.

4 **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.

5 **What are linear and non-linear data Structures?**

Linear: A data structure is said to be linear if its elements form a sequence or a linear list. Examples: Array. Linked List, Stacks and Queues

Non-Linear: A data structure is said to be non-linear if the traversal of nodes is nonlinear in nature. Example: Graph and Trees.

6 **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 O(n) notation.
- Average case is represented by Θ(n) notation.

7 **What are the various operations that can be performed on different Data Structures?**

- Insertion ? Add a new data item in the given collection of data items.
- Deletion ? Delete an existing data item from the given collection of data items.
- Traversal ? Access each data item exactly once so that it can be processed.
- Searching ? Find out the location of the data item if it exists in the given collection of data items.
- Sorting ? Arranging the data items in some order i.e. in ascending or descending order in case of numerical data and in dictionary order in case of alphanumeric data.

8 **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

9 **What is a Linked List and What are its types?**

A linked list is a linear data structure (like arrays) where each element is a separate object. Each element (that is node) of a list is comprising of two items – the data and a reference to the next node.Types of Linked List :

- Singly Linked List : In this type of linked list, every node stores address or reference of next node in list and the last node has next address or reference as NULL. For example 1->2->3->4->NULL
- Doubly Linked List : Here, here are two references associated with each node, One of the reference points to the next node and one to the previous node. Eg. NULL<-1<->2<->3->NULL
- Circular Linked List : Circular linked list is a linked list where all nodes are connected to form a circle. There is no NULL at the end. A circular linked list can be a singly circular linked list or doubly circular linked list. Eg. 1->2->3->1 [The next pointer of last node is pointing to the first].

10 **Can doubly linked be implemented using a single pointer variable in every node?**

Doubly linked list can be implemented using a single pointer.

11 How are linked lists more efficient than arrays?

- Insertion and Deletion

Insertion and deletion process is expensive in an array as the room has to be created for the new elements and existing elements must be shifted.

But in a linked list, the same operation is an easier process, as we only update the address present in the next pointer of a node.

- Dynamic Data Structure

Linked list is a dynamic data structure that means there is no need to give an initial size at the time of creation as it can grow and shrink at runtime by allocating and deallocating memory.

Whereas, the size of an array is limited as the number of items is statically stored in the main memory.

- No wastage of memory

As the size of a linked list can grow or shrink based on the needs of the program, there is no memory wasted because it is allocated in runtime.

In arrays, if we declare an array of size 10 and store only 3 elements in it, then the space for 3 elements is wasted. Hence, chances of memory wastage is more in arrays.

12 What are Infix, prefix and Postfix notations?

Infix notation: X + Y – Operators are written in-between their operands. This is the usual way we write expressions. An expression such as

A * (B + C) / D

Postfix notation (also known as “Reverse Polish notation”): X Y + Operators are written after their operands. The infix expression given above is equivalent to

A B C + * D/

Prefix notation (also known as “Polish notation”): + X Y Operators are written before their operands. The expressions given above are equivalent to

/ A * + B C D

13 What is stack?

In data-structure, stack is an Abstract Data Type (ADT) used to store and retrieve values in Last In First Out method.

14 Why do we use stacks?

Stacks follows LIFO method and addition and retrieval of a data item takes only O(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.

15 What operations can be performed on stacks?

The below operations can be performed on a stack −

- push() − adds an item to stack
- pop() − removes the top stack item
- peek() − gives value of top item without removing it
- isempty() − checks if stack is empty
- isfull() − checks if stack is full

16 What is a queue in data-structure?

A queue is a data structure that can simulate a list or stream of data. In this structure, new elements are inserted at one end, and existing elements are removed from the other end.

17 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

18 Difference between Stack and Queue.

The difference between stacks and queues is in removing. In a stack we remove the item the most recently added; in a queue, we remove the item the least recently added. Both Queues and Stacks can be implemented using Arrays and Linked Lists.

19 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 O(n) and worst-case complexity is O(n2).

20 How to implement a stack using queue?

A stack can be implemented using two queues. Let stack to be

implemented be ‘s’ and queues used to implement be ‘q1’ and ‘q2’. Stack ‘s’ can be implemented in two ways:

- Method 1 (By making push operation costly)
- Method 2 (By making pop operation costly)

21 How to implement a queue using stack?

A queue can be implemented using two stacks. Let queue to be implemented be q and stacks used to implement q be stack1 and stack2. q can be implemented in two ways:

- Method 1 (By making enQueue operation costly)
- Method 2 (By making deQueue operation costly)

22 What is binary search?

A binary search works only on sorted lists or arrays. This search selects the middle which splits the entire list into two parts. First the middle is compared.

This search first compares the target value to the mid of the list. If it is not found, then it takes decision on whether.

23 What is bubble sort and how bubble sort works?

Bubble sort is comparison-based algorithm in which each pair of adjacent elements is compared and elements are swapped if they are not in order. Because the time complexity is O(n2), it is not suitable for large set of data.

24 What is Insertion Sort?

Insertion sort divides the list into two sub-list, sorted and unsorted. It takes one element at time and finds it appropriate location in sorted sub-list and insert there. The output after insertion is a sorted

sub-list. It iteratively works on all the elements of unsorted sub-list and inserts them to sorted sub-list in order.

25 What is selection sort?

Selection sort is in-place sorting technique. It divides the data set into two sub-lists: sorted and unsorted. Then it selects the minimum element from unsorted sub-list and places it into the sorted list. This iterates unless all the elements from unsorted sub-list are consumed into sorted sub-list.

26 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 unsorted sub-list and replaces it with the current element in hand.

27 What is merge sort and how it works?

Merge sort is sorting algorithm based on divide and conquer programming approach. It keeps on dividing the list into smaller sub-list until all sub-list has only 1 element. And then it merges them in a sorted way until all sub-lists are consumed. It has run-time complexity of O(n log n) and it needs O(n) auxiliary space.

28 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 is arranged in the left partition and greater values are

arranged in the right partition. Each partition is recursively sorted using quick sort.

29 Are linked lists considered linear or non-linear data structures?

It depends on where you intend to apply linked lists. If you based it on storage, a linked list is considered non-linear. On the other hand, if you based it on access strategies, then a linked list is considered linear.

30 What is the difference between a PUSH and a POP?

Pushing and popping applies to the way data is stored and retrieved in a stack. A push denotes data being added to it, meaning data is being “pushed” into the stack.

31 What is a binary search tree?

A binary search tree is a binary tree with a special provision where a node's left child must have value less than its parent's value and node's right child must have value greater than it's parent value.

32 How do you insert a new item in a binary search tree?

Assuming that the data to be inserted is a unique value (that is, not an existing entry in the tree), check first if the tree is empty. If it’s empty, just insert the new item in the root node. If it’s not empty, refer to the new item’s key. If it’s smaller than the root’s key, insert it into the root’s left subtree, otherwise, insert it into the root’s right subtree.

33 What is the minimum number of nodes that a binary tree can have?

A binary tree can have a minimum of zero nodes, which occurs when the nodes have NULL values. Furthermore, a binary tree can also have 1 or 2 nodes.

34 What is a tree?

A tree is a minimally connected graph having no loops and circuits.

35 What is a binary tree?

A binary tree has a special condition that each node can have two children at maximum.

36 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

37 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.

Balance Factor = height(left-subtree) – height(right-subtree)

38 What is hashing?

Hashing is a technique to convert a range of key values into a range of indexes of an array. By using hash tables, we can create an

associative data storage where data index can be find by providing its key values.

39 What is a dequeue?

A dequeue is a double-ended queue. This is a structure wherein elements can be inserted or removed from either end.

40 What is FIFO?

FIFO stands for First-in, First-out, and is used to represent how data is accessed in a queue. Data has been inserted into the queue list the longest is the one that is removed first.

Algorithm	Time Complexity		
	Best	Average	Worst
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)
Heap Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Quick Sort	Ω(n log(n))	θ(n log(n))	O(n^2)
Merge Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)
Radix Sort	Ω(nk)	θ(nk)	O(nk)