Ques - What is data structure? Discuss different type of data structure.

Data structure is a storage that is used to store and organize data. It is a way of arranging data on a computer so that it can be accessed and updated efficiently.

Types of Data Structure - Basically, data structures are divided into two categories:

1. Linear data structure
2. Non-linear data structure

Linear data structures - In linear data structures, the elements are arranged in sequence one after the other. Since elements are arranged in particular order, they are easy to implement. However, when the complexity of the program increases, the linear data structures might not be the best choice because of operational complexities.

Array data structure - In an array, elements in memory are arranged in continuous memory. All the elements of an array are of the same type. And the type of elements that can be stored in the form of arrays is determined by the programming language.

Stack data structure - In stack data structure, elements are stored in the LIFO principle. That is, the last element stored in a stack will be removed first.

Example - It works just like a pile of plates were the last plate kept on the pile will be removed first.

Queue data structure - Unlike stack, the queue data structure works in the FIFO principle where first element stored in the queue will be removed first.

Example - It works just like a queue of people in the ticket counter, where first person on the queue will get the ticket first.

Linked List data structure - In linked list data structure, data elements are connected. Through a series of nodes. And each node contains the data items and address to the next node. You have to start somewhere, so we give the address of the first node a special name called HEAD. Also, the last node in the linked list can be identified because its next portion points to NULL.

Non-linear data structures - Unlike linear data structures, elements in non-linear data structures are not in any sequence. Instead, they are arranged in a hierarchical manner where one element will be connected to one or more elements.

Tree data structure - A tree is a nonlinear hierarchical data structure that consists of nodes connected by edges. Similar to a graph, a tree is also a collection of vertices and edges. However, in tree data structure, there can only be one edge between two vertices.

Graph data structure - A graph data structure is a collection of nodes that have data and are connected to other nodes.

Ques – What is circular queue? Why concept of circular queue came.

A circular queue is the extended version of a regular queue where the last element is connected to the first element. Thus, forming a circle-like structure.
The circular queue solves the major limitation of the normal queue. In a normal queue, after a bit of insertion and deletion, there will be non-usable empty space.

Concept - There was one limitation in the array implementation of Queue. If the rear reaches to the end position of the Queue, then there might be possibility that some vacant spaces are left in the beginning which cannot be utilized. So, to overcome such limitations, the concept of the circular queue was introduced.

Overflow condition - In a circular queue, overflow occurs when the queue is full, and we try to insert an element into it. There are two conditions for overflow in a circular queue: (1) front = 0 and rear = capacity - 1; (2) front = rear + 1. If any of these two conditions is satisfied, it means that the circular queue is full.

Ques – What is single linked-list? List different type of operations performed a single linked list.

Singly linked lists contain nodes which have a 'value' field as well as 'next' field, which points to the next node in line of nodes. Operations that can be performed on singly linked lists. include **insertion**, **deletion,** and **traversal**.

Ques – List properties of AVL Tree?

AVL trees are self-balancing binary search trees. Here are some properties of AVL trees:

- **Binary Search Tree**: It follows the general properties of a Binary Search Tree.
- **Balanced Subtrees**: Each subtree of the tree is balanced, i.e., the difference between the height of the left and right subtrees is at most 1.

- **Self-Balancing**: The tree balances itself when a new node is inserted. Therefore, the insertion operation is time-consuming

Ques – Define minimum spanning tree?

A minimum spanning tree (MST) is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. That is, it is a spanning tree whose sum of edge weights is as small as possible.

Ques - What are the disadvantage of representing a linear queue using an array? How are they overcome?

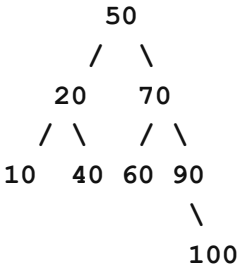
The disadvantages of representing a linear queue using an array are:

- **Static Data Structure** -The size of the queue is fixed.
- **Fixed Size** - If the queue has a large number of enqueue and dequeue operations, at some point (in case of linear increment of front and rear indexes) we may not be able to insert elements in the queue even if the queue is empty (this problem is avoided by using circular queue).

These disadvantages can be overcome by using a circular queue[1]. A circular queue is a data structure that uses a single, fixed-size buffer as if it were connected end-to-end. This structure lends itself easily to buffering data streams

Ques - Construct the binary search tree for the following sequence of numbers.

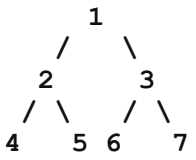
50,70,60,20,90,10,40,100



Ques – Explain tree representation as an array and linked list with suitable examples.

Tree representation as an array - In this representation, we use an array to store the elements of the tree. The root element is stored at index 0, and the left and right children of any node at index i are stored at indices 2i+1 and 2i+2 respectively. This representation is useful when we want to perform operations like searching, insertion, and deletion in the tree.

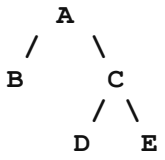
Here's an **Example**: Tree –



Array : [1, 2, 3, 4, 5, 6, 7]

Tree representation as a linked list - In this representation, we use a linked list to store the elements of the tree. Each node in the linked list has a left child pointer and a right child pointer. This representation is useful when we want to perform operations like traversal of the tree. Here's an

Example: Tree -



Linked List: A -> B -> C -> D -> E

Ques – why do we need data structure?

Data structures are important because they allow us to store, organize, and access data efficiently. They are necessary for reasonable processor speeds because the more data a computer must handle, the faster its processor must work. As data increases, the processor can fail to keep up unless that data is well organized into a data structure.

Data structures are also used in software design as major factors for collection, storing and organizing of data rather than algorithms in some programming languages. In almost every software system and program, data structures are often included now-a-days. Often data structures are used in the combination of algorithms. This allows the management of large amounts of data in an efficient way.

Ques – What are the merits and demerits of array implementation over linked list?

Merits of array implementation over linked list-

- Arrays have better cache locality that can make a pretty big difference in performance.
- Arrays allow for random access to elements.
- Arrays are simpler to implement than linked lists.

Demerits of array implementation over linked list-

- Arrays have a fixed size at initialization, and you have to write code to increase the size of the array.
- Insertion and deletion operations are more expensive in arrays than in linked lists because elements need to be shifted around.
- Arrays require contiguous memory allocation which can be difficult or impossible in some cases

Ques – What is linked list? What are different operations that can be performed on the linked list data structure?

A linked list is a linear data structure where each element is a separate object. Each element (we call it a node) of a list is comprising of two items - the data and a reference to the next node. The last node has a reference to null. The entry point into a linked list is called the head of the list. It should be noted that head is not a separate node, but the reference to the first node. If the list is empty, then the head is a null reference.

Here are some operations that can be performed on the linked list data structure:

- **Insertion**: Adds an element at the beginning of the list.
- **Deletion**: Deletes an element at the beginning of the list.
- **Searching**: Searches an element using the given key.
- **Display**: Displays all elements in the linked list.
- **Deleting an element at a given key**: Deletes an element at a given key

Ques – List application of data structure?

Data structures are used in almost every program or software system. Here are some applications of data structures:

- **Arrays**: Used to store and access large amounts of data.
- **Linked Lists**: Used to implement stacks, queues, and other abstract data types.
- **Trees**: Used to implement hierarchical data structures such as file systems and organization charts.
- **Graphs**: Used to represent networks such as social networks and transportation networks.
- **Hash Tables**: Used to implement associative arrays and database indexing.

Ques – Write Characteristics of array.

An array is a collection of similar data types stored in contiguous memory locations. The characteristics of an array are as follows:

- Each element has a similar data type while they may have dissimilar values.
- The whole array is saved contiguously in memory, meaning there are no spaces between elements.
- Array elements are stored in subsequent memory blocks in primary memory.
- Array name represents its base address

Ques – What is linear queue? Write underflow condition for linear queue.

A linear queue is a queue in which the elements are added at one end and removed from the other end. **Overflow condition** -The underflow condition for a linear queue occurs when we try to remove an element from an empty queue.

Ques – List the area of applications where stack data structure can be used.

Stack data structure can be used in various areas such as:

- Function calls and recursion: When a function is called, the current state of the program is pushed onto the stack.
- Undo/Redo operations: The undo-redo feature in various applications uses stacks to keep track of the previous actions.
- Expression evaluation and conversion.
- Syntax parsing.
- Memory management.

Ques – Draw the binary tree whose sequential representation is given below.