1) What do you mean by Data Structures?

Answer: The data structure is a way that specifies how to organize and manipulate the data. It also defines the relationship between them. Some examples of Data Structures are arrays, Linked List, Stack, Queue, etc. Data Structures are the central part of many computer science algorithms as they enable the programmers to handle the data in an efficient way

2) Define The Goals Of Data Structure?

Answer: Goals are that the data is stored in organize manner so that it becomes easy to retrieve that data. To design the algorithm that makes efficient use of computer's resource

3) What is the need of DS?

Answer: Data Structure allows efficient data search and retrieval .so that time complexity and space complexity can be reduced.

4) List out the areas in which Data Structure is used extensively (real time examples)?

- Arrays are used in image processing.
- In the case of music players linked list is used to switch between music.
- E-mails, Google photos' any gallery, YouTube downloads and Scratch card's earned after Google pay transaction are example of stack.
- Uploading and downloading photo's is example of queue.
- The graph is used in GPS navigation system. And also used in Facebook, Instagram and all social media networking sites where every user is Node.

5) List different types of data structures.

I)Linear Data Structure

1. Array
2. Stack
3. Queue
4. Linked List

II)Non Linear Data Structure

1. Tree
2. Graph

6) What Does Abstract Data Type Mean?

The definition of ADT only mentions what operations are to be performed but not how these operations will be implemented. It does not specify how data will be organized in memory and what algorithms will be used for implementing the operations.

7) What is an Recursion?

Answer: Recursion in java is a process in which a method calls itself continuously. A method in java that calls itself is called recursive method. Recursion is the process of defining a problem (or the solution to a problem) in terms of (a simpler version of) itself.

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called a recursive function. Using recursive algorithm, certain problems can be solved quite easily. Examples of such problems are Towers of Hanoi (TOH), Inorder/Preorder/Postorder Tree Traversals, DFS of Graph, etc.

8) List and explain types of recursion :

Answer :Types of Recursions:
Recursion are mainly of two types depending on whether a function calls itself from within itself or more than one function call one another mutually. The first one is called direct recursion and another one is called indirect recursion.

1]. Direct Recursion: These can be further categorized into four types:

1)Tail Recursion: If a recursive function calling itself and that recursive call is the last statement in the function then it's known as Tail Recursion

2) Head Recursion: If a recursive function calling itself and that recursive call is the first statement in the function then it's known as Head Recursion.

3)Tree Recursion: To understand Tree Recursion let's first understand Linear Recursion. If a recursive function calling itself for one time then it's known as Linear Recursion. Otherwise if a recursive function calling itself for more than one time then it's known as Tree Recursion.

4)Nested Recursion: In this recursion, a recursive function will pass the parameter as a recursive call. That means "recursion inside recursion".

2]. Indirect Recursion: In this recursion, there may be more than one functions and they are calling one another in a circular manner.

9) Explain the data structures used to perform recursion?

Answer : Many programming languages implement recursion by means of **stacks**. Generally, whenever a function (**caller**) calls another function (**callee**) or itself as callee, the caller function transfers execution control to the callee. This transfer process may also involve some data to be passed from the caller to the callee.

This implies, the caller function has to suspend its execution temporarily and resume later when the execution control returns from the callee function. Here, the caller function needs to start exactly from the point of execution where it puts itself on hold. It also needs the exact same data

values it was working on. For this purpose, an activation record (or stack frame) is created for the caller function.

10)List the examples where recursion is used

Recursion can be equally well applied to computer algorithms: Some Computer related examples include:
Adding a list of numbers,
Computing the Fibonacci sequence,
computing a Factorial,
and Sudoku.
Real time example :Rotation of fan, Talking to itself, Sea-saw

11) Explain the difference between Recursion and Iteration; justify which to use when, Tail recursion?

Recursion:

- Recursion uses selection structure.
- Infinite recursion occurs if the recursion step does not reduce the problem in a manner that converges on some condition (base case) and Infinite recursion can crash the system.
- Recursion terminates when a base case is recognized.
- Recursion is usually slower than iteration due to the overhead of maintaining the stack.
- Recursion uses more memory than iteration.
- Recursion makes the code smaller.

- **Iteration**
- Iteration uses repetition structure.
- An infinite loop occurs with iteration if the loop condition test never becomes false and infinite looping uses CPU cycles repeatedly.
- Iteration terminates when the loop condition fails.
- Iteration does not use the stack so it's faster than recursion.
- Iteration consumes less memory.
- Iteration makes the code longer

- **Tail recursion**

The tail recursion is basically using the recursive function as the last statement of the function. So when nothing is left to do after coming back from the recursive call, which is called tail recursion. We will see one example of tail recursion.

12) Difference between Primitive and Non Primitive DS

Primitive DS:

Primitive data structure is a kind of data structure that stores the data of only one type

Examples of primitive data structure are integer, character, and float.

Primitive data structure will contain some value, i.e., it cannot be NULL.

Non Primitive DS:

Non-primitive data structure is a type of data structure that can store the data of more than one type.

Examples of non-primitive data structure are Array, Linked list, stack.

Non-primitive data structure can consist of a NULL value.

13) Difference between Linear and Non Linear DS
Answer:
Linear:
- In a linear data structure, data elements are arranged in a linear order where each and every elements are attached to its previous and next adjacent.
-Single level is involved.
- Its implementation is easy in comparison to non-linear data structure.
-Data elements can be traversed in a single run only.
- In a linear data structure, memory is not utilized in an efficient way.
-Ex. array, stack, queue, linked list etc.
- Applications of linear data structures are mainly in application software development.

Non Linear:
- In a non-linear data structure, data elements are attached in hierarchically manner.
- Whereas in non-linear data structure, multiple levels are involved.
- While its implementation is complex in comparison to linear data structure.
- While in non-linear data structure, data elements can't be traversed in a single run only.
- While in a non-linear data structure, memory is utilized in an efficient way.
- While its examples are: trees and graphs.
- Applications of non-linear data structures are in Artificial Intelligence and image processing.

14) What are different characteristics of an Algorithm? types of Algorithm.
Answer:
These are the basic characteristics for every Algorithm

Input specified.

Output specified.

Definiteness.

Effectiveness.
Finiteness.
Independent.

-Types of Algorithms

Bubble Sort, Insertion Sort, Merge Sort, Quick Sort, Selection Sort, AES, Soundex.

15) State Advantages and Dis advantages of Recursion.
Answer:

Advantage:

1.The code may be easier to write.
2. To solve such problems which are naturally recursive such as tower of Hanoi.
3. Reduce unnecessary calling of function.
4. Extremely useful when applying the same solution.
5. Recursion reduce the length of code.
6. It is very useful in solving the data structure problem.
7. Stacks evolutions and infix, prefix, postfix evaluations etc.

Disadvantages of recursion :

1. Recursive functions are generally slower than non-recursive function.
2. It may require a lot of memory space to hold intermediate results on the system stacks.
3. Hard to analyze or understand the code.
4. It is not more efficient in terms of space and time complexity.
5. The computer may run out of memory if the recursive calls are not properly checked.