1. **What is data structure?**

A data structure is a way of organizing data that considers not only the items stored, but also their relationship to each other. Advance knowledge about the relationship between data items allows designing of efficient algorithms for the manipulation of data.

1. **List out the areas in which data structures are applied extensively?**
   1. Compiler Design,
   2. Operating System,
   3. Database Management System,
   4. Statistical analysis package,
   5. Numerical Analysis,
   6. Graphics,
   7. Artificial Intelligence,
   8. Simulation
2. **What are the major data structures used in the following areas : RDBMS, Network data model and Hierarchical data model.**
   1. RDBMS = Array (i.e. Array of structures)
   2. Network data model = Graph
   3. Hierarchical data model = Trees
3. **If you are using C language to implement the heterogeneous linked list, what pointer type will you use?**

The heterogeneous linked list contains different data types in its nodes and we need a link, pointer to connect them. It is not possible to use ordinary pointers for this. So we go for void pointer. Void pointer is capable of storing pointer to any type as it is a generic pointer type.

1. **Minimum number of queues needed to implement the priority queue?**

Two. One queue is used for actual storing of data and another for storing priorities.

1. **What is the data structures used to perform recursion?**

Stack. Because of its LIFO (Last In First Out) property it remembers its 'caller' so knows whom to return when the function has to return. Recursion makes use of system stack for storing the return addresses of the function calls.

Every recursive function has its equivalent iterative (non-recursive) function. Even when such equivalent iterative procedures are written, explicit stack is to be used.

1. **What are the notations used in Evaluation of Arithmetic Expressions using prefix and postfix forms?**

Polish and Reverse Polish notations.

1. **Convert the expression ((A + B) \* C - (D - E) ^ (F + G)) to equivalent Prefix and Postfix notations.**
   1. **Prefix Notation:** - \* +ABC ^ - DE + FG
   2. **Postfix Notation:** AB + C \* DE - FG + ^ -
2. **Sorting is not possible by using which of the following methods? (Insertion, Selection, Exchange, Deletion)**

**Sorting is not possible in Deletion.** Using insertion we can perform insertion sort, using selection we can perform selection sort, using exchange we can perform the bubble sort (and other similar sorting methods). But no sorting method can be done just using deletion.

1. **What are the methods available in storing sequential files ?**
   1. Straight merging,
   2. Natural merging,
   3. Polyphase sort,
   4. Distribution of Initial runs.
2. **List out few of the Application of tree data-structure?**
   1. The manipulation of Arithmetic expression,
   2. Symbol Table construction,
   3. Syntax analysis.
3. **List out few of the applications that make use of Multilinked Structures?**
   1. Sparse matrix,
   2. Index generation.
4. **In tree construction which is the suitable efficient data structure? (Array, Linked list, Stack, Queue)**

Linked list is the suitable efficient data structure.

1. **What is the type of the algorithm used in solving the 8 Queens problem?**

Backtracking.

1. **In an AVL tree, at what condition the balancing is to be done?**

If the 'pivotal value' (or the 'Height factor') is greater than 1 or less than -1.

1. **What is the bucket size, when the overlapping and collision occur at same time?**

One. If there is only one entry possible in the bucket, when the collision occurs, there is no way to accommodate the colliding value. This results in the overlapping of values.

1. **Classify the Hashing Functions based on the various methods by which the key value is found.**
   1. Direct method,
   2. Subtraction method,
   3. Modulo-Division method,
   4. Digit-Extraction method,
   5. Mid-Square method,
   6. Folding method,
   7. Pseudo-random method.
2. **What are the types of Collision Resolution Techniques and the methods used in each of the type?**
   1. **Open addressing (closed hashing),**The methods used include: Overflow block.
   2. **Closed addressing (open hashing),**The methods used include: Linked list, Binary tree.
3. **In RDBMS, what is the efficient data structure used in the internal storage representation?**

B+ tree. Because in B+ tree, all the data is stored only in leaf nodes, that makes searching easier. This corresponds to the records that shall be stored in leaf nodes.

1. **What is a spanning Tree?**

A spanning tree is a tree associated with a network. All the nodes of the graph appear on the tree once. A minimum spanning tree is a spanning tree organized so that the total edge weight between nodes is minimized.

1. **Does the minimum spanning tree of a graph give the shortest distance between any 2 specified nodes?**

No. The Minimal spanning tree assures that the total weight of the tree is kept at its minimum. But it doesn't mean that the distance between any two nodes involved in the minimum-spanning tree is minimum.

1. **Which is the simplest file structure? (Sequential, Indexed, Random)**

Sequential is the simplest file structure.

1. **Whether Linked List is linear or Non-linear data structure?**

According to Access strategies Linked list is a linear one.   
According to Storage Linked List is a Non-linear one.

***What is a Data Structure?***  
A data structure is a way of organizing the data so that the data can be used efficiently. Different kinds of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. For example, B-trees are particularly well-suited for implementation of databases, while compiler implementations usually use hash tables to look up identifiers. (Source: [Wiki Page](http://en.wikipedia.org/wiki/Data_structure))

***Which data structures are used for BFS and DFS of a graph?***  
[Queue is used for BFS](http://www.geeksforgeeks.org/breadth-first-traversal-for-a-graph/) and Stack is used for DFS. [DFS can also be implemented using recursion](http://www.geeksforgeeks.org/depth-first-traversal-for-a-graph/) (Note that recursion also uses function call stack).

***Can doubly linked be implemented using a single pointer variable in every node?***  
Doubly linked list can be implemented using a single pointer. See [XOR Linked List – A Memory Efficient Doubly Linked List](http://www.geeksforgeeks.org/xor-linked-list-a-memory-efficient-doubly-linked-list-set-1/)

***How to implement a stack using queue?***  
See [Implement Stack using Queues](http://www.geeksforgeeks.org/implement-stack-using-queue/)

***How to implement a queue using stack?***  
See [Implement Queue using Stacks](http://www.geeksforgeeks.org/queue-using-stacks/)

***Linked List Questions***  
[Linked List Insertion](http://quiz.geeksforgeeks.org/linked-list-set-2-inserting-a-node/), [Linked List Deletion](http://quiz.geeksforgeeks.org/linked-list-set-3-deleting-node/), [middle of a given linked list](http://www.geeksforgeeks.org/write-a-c-function-to-print-the-middle-of-the-linked-list/), [Nth node from the end of a Linked List](http://www.geeksforgeeks.org/nth-node-from-the-end-of-a-linked-list/),

***Tree Traversal Questions***  
[Inorder, Preorder and Postoder Traversals](http://www.geeksforgeeks.org/618/), [Level order traversal](http://www.geeksforgeeks.org/level-order-tree-traversal/), [Height of Binary Tree](http://www.geeksforgeeks.org/write-a-c-program-to-find-the-maximum-depth-or-height-of-a-tree/).

***How to check if a given Binary Tree is BST or not?***  
If inorder traversal of a binary tree is sorted, then the binary tree is BST. The idea is to simply do inorder traversal and while traversing keep track of previous key value. If current key value is greater, then continue, else return false. See [A program to check if a binary tree is BST or not](http://www.geeksforgeeks.org/a-program-to-check-if-a-binary-tree-is-bst-or-not/) for more details.

**Convert a DLL to Binary Tree in-place**  
See [In-place conversion of Sorted DLL to Balanced BST](http://www.geeksforgeeks.org/in-place-conversion-of-sorted-dll-to-balanced-bst/)

***Convert Binary Tree to DLL in-place***  
See [Convert a given Binary Tree to Doubly Linked List | Set 1](http://www.geeksforgeeks.org/in-place-convert-a-given-binary-tree-to-doubly-linked-list/), [Convert a given Binary Tree to Doubly Linked List | Set 2](http://www.geeksforgeeks.org/convert-a-given-binary-tree-to-doubly-linked-list-set-2/)

***Delete a given node in a singly linked list***  
[Given only a pointer to a node to be deleted in a singly linked list, how do you delete it?](http://www.geeksforgeeks.org/in-a-linked-list-given-only-a-pointer-to-a-node-to-be-deleted-in-a-singly-linked-list-how-do-you-delete-it/)

***Reverse a Linked List***  
[Write a function to reverse a linked list](http://www.geeksforgeeks.org/write-a-function-to-reverse-the-nodes-of-a-linked-list/)

***Detect Loop in a Linked List***  
[Write a C function to detect loop in a linked list](http://www.geeksforgeeks.org/write-a-c-function-to-detect-loop-in-a-linked-list/).

***Which data structure is used for dictionary and spell checker?***  
[Data Structure for Dictionary and Spell Checker?](http://www.geeksforgeeks.org/data-structure-dictionary-spell-checker/)

**Which Data Structure Should be used for LRU cache?**  
[How to implement LRU caching scheme? What data structures should be used?](http://www.geeksforgeeks.org/implement-lru-cache/)

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be **LIFO(Last In First Out) or FILO(First In Last Out).**

**Basic operations :**

**Push:**Adds an item in the stack. If the stack is full, then it is said to be an Overflow condition. (Top=Top+1)**Pop:** Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.(Top=Top-1) **Peek:**Get the topmost item.

**Infix, prefix, 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 YOperators are written before their operands. The expressions given above are equivalent to

/ \* A + B C D

Converting between these notations: [Click here](http://quiz.geeksforgeeks.org/stack-set-2-infix-to-postfix/)

**Tower of Hanoi** is a mathematical puzzle where we have three rods and n disks. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1) Only one disk can be moved at a time.

2) Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.

3) No disk may be placed on top of a smaller disk.

For n disks, total 2n – 1 moves are required

Time complexity : O(2n) [exponential time]

[**Queues**](http://quiz.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/)

Queue is a linear structure which follows a particular order in which the operations are performed. The order is First In First Out **(FIFO)**.  A good example of queue is any queue of consumers for a resource where the consumer that came first is served first. **Stack :** Remove the item the **most recently** added **Queue:** Remove the item the**least recently** added **Operations on Queue:**

**Enqueue:** Adds an item to the queue. If the queue is full, then it is said to be an Overflow condition.

**Dequeue**: Removes an item from the queue. The items are popped in the same order in which they are pushed. If the queue is empty, then it is said to be an Underflow condition.

**Front:** Get the front item from queue.

**Rear**: Get the last item from queue.

[Linked Lists](http://quiz.geeksforgeeks.org/linked-list-set-1-introduction/)

Linked List is a linear data structure. Unlike arrays, linked list elements are not stored at contiguous location; the elements are linked using pointers.

**Advantages over arrays** 1) Dynamic size 2) Ease of insertion/deletion **Drawbacks:** 1) Random access is not allowed. We have to access elements sequentially starting from the first node. So we cannot do binary search with linked lists. 2) Extra memory space for a pointer is required with each element of the list. Representation in C: A linked list is represented by a pointer to the first node of the linked list. The first node is called head. If the linked list is empty, then value of head is NULL. Each node in a list consists of at least two parts: 1) data 2) pointer to the next node In C, we can represent a node using structures. Below is an example of a linked list node with an integer data.

// A linked list node

struct node

{

int data;

struct node \*next;

};