DATA STRUCTURES VIVA QUESTIONS

1Q) What is a Data Structure?

Ans) A **Data Structure** is a data object together with the relationships that exists among the instances & among the individual elements that compose an instance.

2Q) Types of Data Structures and give examples?

Ans) There are two types of Data Structures:

1. **Linear Data Structures:** A data structure is said to be linear if the elements form a sequence. It is sequential and continues in nature i.e. access the data in sequential manner.

   In linear data structure we can not insert an item in middle place and it maintains a linear relationship between its elements

   egs: Array, Linked list, Stack, Queue, Dequeue etc.

2. **Non Linear Data Structures:** A data structure is said to be non-linear if elements do not form a sequence. (Not sequential).

   It does not maintain any linear relationship between their elements. Every data item is attached to several other data items in a way that is specific for reflecting relationships. The data items are not arranged in a sequential structure.

   egs: Trees, Graphs.

[A data structure is linear if every item is related with next and previous item and it is non linear if it is attach with many of the items in specific ways to reflect relationship.]

3Q) What is a Singly Linked List?

Ans) Singly Linked List is a Sequence of dynamically allocated Storage elements, each element of which contains a pointer to its successor. A pointer to the first element of the list is called as head and a pointer to the last element of the list is called as tail used to keep track of the list elements.

4Q) What is Doubly Linked List?

Ans) In Doubly Linked List each element contains two pointers: One Pointer points to its successor and another to its predecessor (previous element).It is also called as two way linked list (traversing can be done in both directions).

5Q) Differentiate Array and Linked List?

Ans)

Array	Linked List
1.Size of the array is fixed	1.Size of the linked list is not fixed
2. Memory is allocated Statically (or) Dynamically (at run time). (If the memory is allocated for an array Statically(at compile time) it is called Static Array and if memory is allocated at run time (dynamically)using operator new it is called Dynamic Array)	2. Memory is allocated dynamically (at runtime).
3.Memory wastage will be there if all the array positions are not utilized	3.Memory is not wasted as only Required memory is allocated

STACKS: (LIFO DATA STRUCTURE)

6Q) What is a Stack? (LIFO Data Structure)

Ans) Stack is an ordered collection of items into which items can be inserted and deleted from only one end called as "Top" of the Stack. It is also called as LIFO list.(Last In First Out).

7Q) What is Stack Underflow?

Ans) Is Stack is empty and POP operation is performed it is not possible to delete the items. This situation is called Stack Underflow.

8Q) What is Stack Overflow?

Ans) If Stack is full and PUSH operation is performed it is not possible to insert or Push the new items into the stack. This situation is called Stack Overflow.

9Q) What are the Applications of Stack?

Ans) i) Stacks are used to convert Infix expression into Postfix.

ii) Stacks are used to Evaluate Postfix Expression.

iii) Stacks are used in recursion etc.

10Q) What is the use of Postfix expressions?

Ans) Postfix Expressions are easy to evaluate as postfix expressions does not make use of operator precedence not does it require the use of parenthesis.

QUEUES: (FIFO DATA STRUCTURE)

11Q) What is a Queue?

Ans) It is an ordered collection of items into which items can be inserted from one end called as REAR end and items are deleted from other end called as FRONT end of the Queue. It is also called as FIRST IN FIRST OUT (FIFO) LIST.

12Q) What are the applications of Queues?

Ans) i) Queues are used in Breadth First Traversal of a Tree.

ii) Queues are used in implementation of Scheduling algorithms of Operating Systems.

13Q) What is a Circular Queue?

Ans) In Circular Queue, the first position of the array is kept behind the last position of the array.

14Q) Differentiate Linear Queue and Circular Queue?

Ans) In Linear Queue once the queue is full and the deletion is performed, even if first position is free(vacant) it is not possible to insert the item in that position whereas in Circular Queue it is possible since the first position is kept behind the last position.

15Q) What is Dequeue? (Double Ended Queue)

Ans) In Double Ended Queue insertion and deletion are possible from both the ends.

TREES:

16Q) What is a Tree?

Ans) Tree is a finite non-empty set of nodes with the following properties:

i) A designated node of the set is called as root of the tree and
ii) The remaining nodes are partitioned into n>=0 subsets, each of which is a tree.

Degree of a node: The number of sub trees attached to a node is called degree of that node and the maximum degree of any node in a tree is called **degree of that tree.**

[Note: In a general tree degree of a node is not fixed]

Nodes that have degree zero are called **Leaf or Terminal Nodes**. Consequently, the other nodes are referred to as **Non-Terminals.**

The **Level of a node** is defined by letting the root be at level o or 1.

The **height or depth of a tree** is defined to be the maximum level of any node in the tree.

17Q) What is a Binary Tree?

Ans) A Binary tree T is a finite set of nodes with the following properties:

i) Either the set is empty, T=Ø or
ii) The set consists of a root and exactly two distinct binary trees TL and TR,T={r,TL,TR}.TL is the left subtree and TR is the right subtree of T.

[Note: Maximum degree of any node in a binary tree is 2.Degree of a node in a Binary Tree be either 0 or 1 or 2]

18Q) What is Tree Traversal? List different Tree Traversal Techniques?

Ans) Visiting all nodes of a tree exactly once in a systematic way is called Tree Traversal. Different types of tree traversals are

i) **Depth First Traversals**: PREOREDER (N L R) ,INORDER (L N R) & POSTORDER (L R N) Traversals.
ii) **Breadth First Traversal** (or) **Level Order Traversal** (Visiting Level by level from left to right)

19Q) What is a Binary Search Tree? Give one example?

Ans) A Binary Search Tree T is a finite set of keys. Either the set is empty T=Ø ,or the set consists of a root "r" and exactly two binary search trees TL and TR,T = {r,TL,TR} with the following properties:

i) All the keys contained in the left subtree are less than the root key.
ii) All the keys contained in the right subtree are larger than the root key.

[Note: Duplicates are not allowed in a Binary Search Tree]

20Q) What is the best, average and worst case time complexity of insertion, deletion and Search operations in a Binary Search Tree?

Ans) In Best and Avg case **O(log n)** and in Worst case **O(n).**

21Q) What is an AVL Search Tree? What is AVL Balance Condition?

Ans) An AVL Search Tree is a balanced binary search tree. An empty binary tree is AVL balanced. A non – empty binary tree, T={r,TL,TR} is AVL balanced if both TL & TR are AVL balanced and $| h_L - h_R | <= 1$,

Where : h_L is the Height of the left subtree and h_R is the Height of the right subtree.

[**Note** : Allowable balance factors of any node is an AVL tree are 0 or 1 or -1 and the use of balancing a binary search tree is even in worst case the time complexity of insert, delete and search operations is reduced to **O(log n)** from **O(n)**]

22Q) What is a Full (perfect) Binary Tree?

Ans) If a binary tree of height 'h' has exactly **[2^{h+1} -1]** nodes then that binary tree is called as Full Binary Tree.

23Q) What is a Complete Binary Tree?

Ans) Complete Binary Tree is a binary tree T = {r,TL,TR} with the following properties:

If "i" is the index of any node in a complete binary tree then:

i) The parent of "i" is at position "i/2" [if i=1 it is the root node and has no parent]
ii) The left child of node "i" is at position "2i" [if 2i>n then no left child exists]
iii) The right child of node "i" is at position "2i+1" [if 2i+1>n then no right child exists]

[Note: In a complete binary tree nodes are filled level by level from left to right]

24Q) List one application of trees (Search trees)?

Ans) Search trees are used to implement dictionaries.

PRIORITY QUEUES:

25Q) What is Priority Queue and differentiate Priority Queue and Queue?

Ans) In Priority Queue each item is associated with a priority.

In Priority Queue items can be inserted arbitrary order but the items are deleted based upon the priority that is the item with highest priority is deleted first. Whereas in Queue the items are inserted from rear end and deleted from front end and the item which is inserted first is deleted first (FIFO).

26Q) What is a Min Heap?

Ans) Min Heap is a Complete Binary Tree in which the key value at parent is always less than or equal to its child values.

27Q) What is a Max Heap?

Ans) In Max Heap the key value at parent is always larger or equal to its child values.

28Q) What is a (Binary) Heap?

Ans) A (Binary) Heap is a Complete Binary Tree with Heap Ordered Property (Min Heap or Max Heap) **[Note: Duplicates are allowed in a Heap]**

GRAPHS:

29Q) What is a Graph? Name various Graph Representation Techniques?

Ans) A Graph G = (V,E) is Set of Vertices and Edges.

A Graph can be represented as an Adjacency Matrix (or) as an Adjacency List.

30Q) What is difference between a Graph and a Tree?

Ans) A Graph contains Cycles (loops) but a Tree does not contain any cycles.

[A Tree contains a root node but Graph does not contain the root node.]

31Q) What is a Spanning Tree?

Ans) A Spanning Tree T = (V', E') is a Sub Graph of G = (V,E) with following properties:

i) V = V' [The vertices in a graph and spanning tree are same]
ii) T is Acyclic.
iii) T is connected.

[Note: I f Graph has "n" vertices the Spanning Tree contains exactly "n - 1" edges]

32Q) Name the methods to construct a Minimum cost Spanning Tree?

Ans) Prim's method and Kruskal's method.

SORTING & SEARCHING:

33Q) What is Linear Search? What is the time complexity of searching an item in a list using linear search?

Ans) In linear search the item to be searched is compared with each item starting with the item in the first position of the array until the item is found or all the items of the array. The time complexity of linear search in Worst case is **O(n).**

34Q) What is Binary Search method? What is the time complexity?

Ans) Binary Search method can be used only if the list is Sorted. In binary search method first the array is divided in to two by finding the mi position of the array and the item to be searched is compared with the mid value.

- If the item to be searched is less than the mid value, the item is searched in the left subarray.
- If the item is larger than the mid value it is searched in the right sub array recursively.
- Otherwise it equals to the mid value and search is successful.
- The time complexity of Binary Search method is **O(log n).**

35Q) What is Sorting?

Ans) Arranging the elements in (ascending or descending) some particular order is called Sorting.

36Q) Study the procedure of following Sorting techniques?

Ans) i) MERGE SORT ii) QUICK SORT iii) INSERTION SORT iv) SELECTION SORT v) HEAP SORT

37Q) Time complexities of Sorting Techniques:

SORTING TECHNIQUE	BEST CASE	AVG CASE	WORST CASE
INSERTION SORT	**O(n)**	**$O(n^2)$**	**$O(n^2)$**
SELECTION SORT	**$O(n^2)$**	**$O(n^2)$**	**$O(n^2)$**
MERGE SORT	**O(n log n)**	**O(n log n)**	**O(n log n)**
QUICK SORT	**O(n log n)**	**O(n log n)**	**$O(n^2)$**
HEAP SORT	**O(n log n)**	**O(n log n)**	**O(n log n)**

[NOTE: If the given array is already sorted then Insertion Sort technique is best to sort the given sequence.]

Study the procedures of all the sorting methods.

38Q) Comparison between Quick Sort and Heap Sort?

Ans) If the size of the array to be sorted is very large the Heap sort is efficient than Quick sort since in worst case the time complexity of Heap sort remains same as O (n log n) but Quick Sort time complexity changes to O (n^2).But if the size of the Array is small then Quick sort is efficient than Heap sort.

HASHING:

39Q) What is Hashing?

Ans) Hashing is used to determine the position of a Key by using the Key itself. To determine the position of the key it makes use of Function called Hash function.

A **hash function** takes a group of characters (called a key) and maps it to a value of a certain length (called a hash value or hash). The hash value is representative of the original string of characters, but is normally smaller than the original.

(or)

A **hash function** maps a key into a bucket in the hash table.

40Q) List different Hashing methods.

Ans) i) Division method ii) Mid Square method iii) Folding method iv) Digit Analysis method

- **Hashing Methods**
  There are eight hashing methods they are:

1. **Modulo-division**
2. **Midsquare**
3. **Digit Extraction (or) Digit Analysis**
4. **Folding**

1. **Modulo-division Method:** This is also known as Division Remainder method.
   This Hash function assumes the keys are non-negative integers.The home bucket is obtaind by using the modulo(%) operator.
   The key is divided by some number D and the remainder is used as the home bucket for k.
   h(k) = k % D
   This function gives bucket addresses in the range 0 through D-1,so the hash table must have at least b = D buckets.

(or)

- This algorithm works with any list size, but a list size that is a prime number produces fewer collisions than other list sizes.
- The formula to calculate the address is:
  Address = key MODULO listsize + 1
  Where listsize is the number of elements in the arrary.
  Example:
  Given data : Keys are : 137456 214562 140145
  137456 % 19 +1 = 11, 214562 % 19 + 1 = 15, 140145 % 19 + 1 = 2

2. **Midsquare Method**
   The Mid – square hash function determines the home bucket for a key by squaring the key and then using an appropriate number of bits from the middle of the square to obtain the bucket address; the key is assumed to be an integer.

   (or)

- In midsquare hashing the key is squared and the address is selected from the middle of the square number.
- Limitation is the size of the key.
  Example:
  9452^2 = 89340304: address is 3403

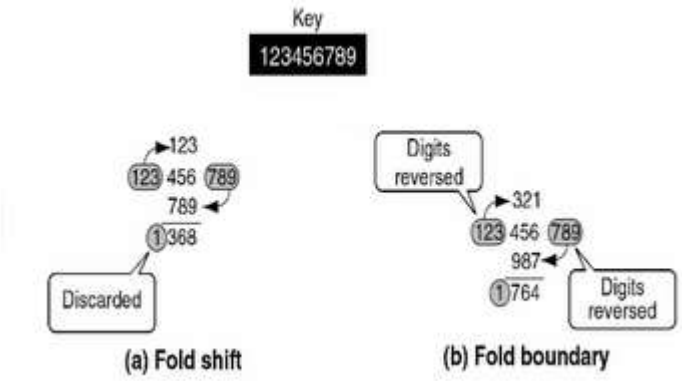
3. **Folding Method**
   Two folding methods are used they are:

i) Fold shift
ii) Fold boundary

   i) Fold Shift
   In fold shift the key value is divided into parts whose size matches the size of the required address. Then the left and right parts are shifted and added with the middle part.

   ii) Fold boundary
   In fold boundary the left and right numbers are folded on a fixed boundary between them and the center number. The two outside values are thus reversed.

   Example:



4. **Digit-extraction (or) Digit Analysis Method:** Using digit extraction selected digits are extracted from the key and used as the address.

   Example: Using six-digit employee number to hash to a three digit address (000-999), we could select the first, third, and fourth digits(from the left) and use them as the address.

   The keys are: 379452 -> 394, 121267 -> 112, 378845 -> 388

41Q) What is Collision?

Ans) By using the hash function if two different keys are hashed to the same position, this situation is called Collision. That is: If x and y are two different ($x \neq y$) and $h(x) = h(y)$ this situation is called Collision.

- A collision occurs when the home bucket for a new pair is occupied by a pair with a different key.
- An overflow occurs when there is no space in the home bucket for the new pair.
- When a bucket can hold only one pair, collisions and overflows occur together.
- Need a method to handle overflows.

Uniform Hash Function: A uniform hash function maps the keys in keySpace into buckets such that approximately the same number of keys get mapped into each bucket.

OVERFLOW HANDLING (OR) COLLISION RESOLUTION TECHNIQUES

- An overflow occurs when the home bucket for a new pair (key, element) is full.
- We may handle overflows by:
  - **Open Addressing:** Search the hash table in some systematic fashion for a bucket that is not full.
    - Linear probing (linear open addressing).
    - Quadratic probing.
    - Rehashing.
    - Random probing.
  - **Chaining:** Eliminate overflows by permitting each bucket to keep a list of all pairs for which it is the home bucket.
    - Array linear list.
    - Chain.

42Q) List different collision resolution strategies?

Ans) There are several methods for handling collisions, each of them independent of the hashing algorithm.

1.Open Addressing : a) Linear Probing b)Quadratic Probing c) Rehashing d) Random Probing

2. Chaining

[Note: Primary clustering occurs in Linear Probing and it is avoided in Quadratic Probing and Double Hashing (Rehashing)]

Linear Probe : In Linear Probing, when inserting a new pair whose key is k, we search the hash table bucket in the order, ht[h(k) + i] % b,0<= i <= b-1,where h is the hash function and b is the number of buckets.

It is a scheme in computer programming for resolving hash collisions of values of hash functions by sequentially searching the hash table for a free location.

Quadratic Probing: In Quadratic Probing, a quadratic function of I is used as the increment. The search is carried out by examining buckets h(k) , $[h(k) + i^2]$ % b and $[h(k) – i^2]$ % b for 1<= I <= (b-1)/2.

It is a scheme in computer programming for resolving collisions in hash tables. It is an open addressing method to handle overflows after a collision takes place in some bucket of a hash table. Quadratic probing operates by taking the original hash value and adding successive values of an arbitrary quadratic polynomial to the starting value.

Random Probing: In Random Probing, the search for a key , k in a hash table with b buckets is carried out by examining the buckets in the order h(k), [h(k) + s(i)] % b, 1<= I <= b-1 where s(i) is a pseudo random number. The random number generator must satisfy the property that every number from 1 to b-1 must be generated exactly once as i ranges from 1 to b-1.

Double hashing: It is a computer programming technique used in hash tables to resolve hash collisions, cases when two different values to be searched for produce the same hash key. It is a popular collision-resolution technique in open-addressed hash tables.

43Q) List one application of Hashing.

Ans) Hashing is used to construct the symbol table used by the language compilers.

44Q) What is the use of Friend Function?

Ans) To access the Private members of one class into another class Friend function is used.