# CS 8391- DATA STRUCTURES

# 2 Marks With Answer

### UNIT I – LINEAR STRUCTURES -LIST

**1. Define Data Structures**

      Data Structures is defined as the way of organizing all data items that consider not only the elements stored but also stores the relationship between the elements.

**2. Define Linked Lists**

     Linked list consists of a series of structures, which are not necessarily adjacent in memory. Each structure contains the element and a pointer to a structure containing its successor. We call this theNext Pointer. The last cell'sNext pointer points to NULL.
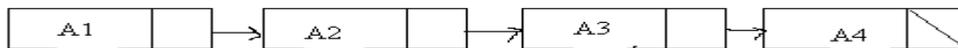


**3. State the different types of linked lists**

      The different types of linked list include singly linked list, doubly linked list and circular linked list.

**4. List the basic operations carried out in a linked list**

    The basic operations carried out in a linked list include:

- Creation of a list
- Insertion of a node
- Deletion of a node
- Modification of a node
- Traversal of the list

**5. List out the advantages of using a linked list**

- It is not necessary to specify the number of elements in a linked list during its declaration

• Linked list can grow and shrink in size depending upon the insertion and deletion that occurs in the list

• Insertions and deletions at any place in a list can be handled easily and efficiently

• A linked list does not waste any memory space

**6. List out the disadvantages of using a linked list**

• Searching a particular element in a list is difficult and time consuming

• A linked list will use more storage space than an array to store the same number of elements

**7. List out the applications of a linked list**

Some of the important applications of linked lists are manipulation of polynomials, sparse matrices, stacks and queues.

**8. State the difference between arrays and linked lists**

| Arrays | Linked Lists |
|---|---|
| Size of an array is fixed | Size of a list is variable |
| It is necessary to specify the number of elements during declaration. | It is not necessary to specify the number of elements during declaration |
| Insertions and deletions are somewhat difficult | Insertions and deletions are carried out easily |
| It occupies less memory than a linked list for the same number of elements | It occupies more memory |

**9. Define an Abstract Data Type (ADT)**

An abstract data type is a set of operations. ADTs are mathematical abstractions; nowhere in an ADT's definition is there any mention of how the set of operations is implemented. Objects such as lists, sets and graphs, along with their operations can be viewed as abstract data types.

**10. What are the advantages of modularity?**

• It is much easier to debug small routines than large routines

• It is easier for several people to work on a modular program simultaneously

• A well-written modular program places certain dependencies in only one routine, making changes easier

## 11. What are the objectives of studying data structures?

• To identify and create useful mathematical entities and operations to determine what classes of problems can be solved using these entities and operations

• To determine the representation of these abstract entities and to implement the abstract operations on these concrete representation

## UNIT II -  LINEAR DATA STRUCTURES – STACKS, QUEUES

1. **What are the types of queues?**

   • Linear Queues – The queue has two ends, the front end and the rear end. The rear end is where we insert elements and front end is where we delete elements. We can traverse in a linear queue in only one direction ie) from front to rear.

   • Circular Queues – Another form of linear queue in which the last position is connected to the first position of the list. The circular queue is similar to linear queue has two ends, the front end and the rear end. The rear end is where we insert elements and front end is where we delete elements. We can traverse in a circular queue in only one direction ie) from front to rear.

   • Double-Ended-Queue – Another form of queue in which insertions and deletions are made at both the front and rear ends of the queue.

**2. List the applications of stacks**

   • Towers of Hanoi

   • Reversing a string

   • Balanced parenthesis

   • Recursion using stack

   • Evaluation of arithmetic expressions

**3.. List the applications of queues**

   • Jobs submitted to printer

   • Real life line

   • Calls to large companies

   • Access to limited resources in Universities

   • Accessing files from file server

**4. Define a stack**

   Stack is an ordered collection of elements in which insertions and deletions are restricted to one end. The end from which elements are added and/or removed is referred

to as top of the stack. Stacks are also referred as piles, push-down lists and last-in-first-out (LIFO) lists.

**5. List out the basic operations that can be performed on a stack**

The basic operations that can be performed on a stack are

• Push operation

• Pop operation

• Peek operation

• Empty check

• Fully occupied check

**6. State the different ways of representing expressions**

The different ways of representing expressions are

• Infix Notation

• Prefix Notation

• Postfix Notation

**7. State the rules to be followed during infix to postfix conversions**

• Fully parenthesize the expression starting from left to right. During

parenthesizing, the operators having higher precedence are first parenthesized

• Move the operators one by one to their right, such that each operator replaces

their corresponding right parenthesis

• The part of the expression, which has been converted into postfix is to be

treated as single operand

**8. Mention the advantages of representing stacks using linked lists than arrays**

• It is not necessary to specify the number of elements to be stored in a stack during its

declaration, since memory is allocated dynamically at run time when an element is added

to the stack

• Insertions and deletions can be handled easily and efficiently

• Linked list representation of stacks can grow and shrink in size without wasting

memory space, depending upon the insertion and deletion that occurs in the list

• Multiple stacks can be represented efficiently using a chain for each stack

**9. Mention the advantages of representing stacks using linked lists than arrays**

• It is not necessary to specify the number of elements to be stored in a stack during its declaration, since memory is allocated dynamically at run time when an element is added to the stack

• Insertions and deletions can be handled easily and efficiently

• Linked list representation of stacks can grow and shrink in size without wasting memory space, depending upon the insertion and deletion that occurs in the list

• Multiple stacks can be represented efficiently using a chain for each stack

**10. Define a queue**

Queue is an ordered collection of elements in which insertions are restricted to one end called the rear end and deletions are restricted to other end called the front end. Queues are also referred as First-In-First-Out (FIFO) Lists.

**11.  Define a priority queue**

Priority queue is a collection of elements, each containing a key referred as the priority for that element. Elements can be inserted in any order (i.e., of alternating priority), but are arranged in order of their priority value in the queue. The elements are deleted from the queue in the order of their priority (i.e., the elements with the highest priority is deleted first). The elements with the same priority are given equal importance and processed accordingly.

**12. State the difference between queues and linked lists**

The difference between queues and linked lists is that insertions and deletions may occur anywhere in the linked list, but in queues insertions can be made only in the rear end and deletions can be made only in the front end.

**13.. Define a Deque**

Deque (Double-Ended Queue) is another form of a queue in which insertions and deletions are made at both the front and rear ends of the queue. There are two variations of a deque, namely, input restricted deque and output restricted deque. The input
restricted deque allows insertion at one end (it can be either front or rear) only. The

output restricted deque allows deletion at one end (it can be either front or rear)

6

**14. What is the need for Priority queue?**

In a multiuser environment, the operating system scheduler must decide which of the several processes to run only for a fixed period of time. One algorithm uses queue. Jobs are initially placed at the end of the queue. The scheduler will repeatedly take the first job on the queue, run it until either it finishes or its time limit is up and place it at the end of the queue if it does not finish. This strategy is not appropriate, because very short jobs will soon to take a long time because of the wait involved in the run.

Generally, it is important that short jobs finish as fast as possible, so these jobs should have precedence over jobs that have already been running. Further more, some jobs that are not short are still very important and should have precedence. This particular application seems to require a special kind of queue, known as priority queue. Priority queue is also called as Heap or Binary Heap.

# UNIT III – NON LINEAR DATA STRUCTURES – TREES

## 1. Define a tree

A tree is a collection of nodes. The collection can be empty; otherwise, a tree consists of a distinguished node r, called the root, and zero or more nonempty (sub) trees T1, T2,…,Tk, each of whose roots are connected by a directed edge from r.
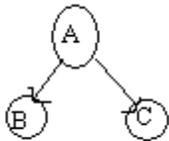
## 2. Define root

This is the unique node in the tree to which further sub-trees are attached.
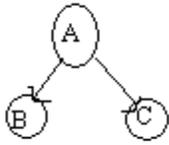


Here, A is the root.

## 3. Define degree of the node

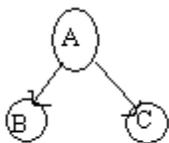The total number of sub-trees attached to that node is called the degree of the node.



For node A, the degree is 2 and for B and C, the degree is 0.
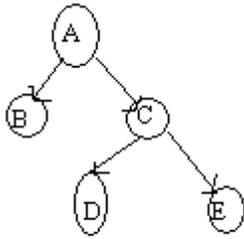
## 4. Define leaves

These are the terminal nodes of the tree. The nodes with degree 0 are always the leaves.



Here, B and C are leaf nodes.
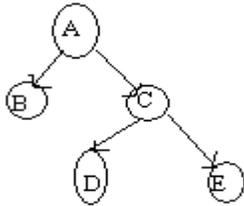
### 5. Define internal nodes

The nodes other than the root and the leaves are called internal nodes.



Here, C is the internal node.

### 6. Define parent node

The node which is having further sub-branches is called the parent node of those sub-branches.



Here, node C is the parent node of D and E

### 7. Define depth and height of a node

For any node ni, the depth of ni is the length of the unique path from the root to ni. The height of ni is the length of the longest path from ni to a leaf.

### 8. Define depth and height of a tree

The depth of the tree is the depth of the deepest leaf. The height of the tree is equal to the height of the root. Always depth of the tree is equal to height of the tree.

### 9. What do you mean by level of the tree?

The root node is always considered at level zero, then its adjacent children are supposed to be at level 1 and so on.

Here, node A is at level 0, nodes B and C are at level 1 and nodes D and E are at level 2.
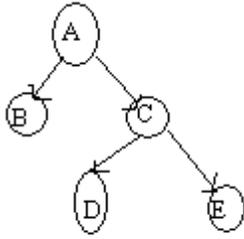
### 10. Define forest

A tree may be defined as a forest in which only a single node (root) has no predecessors. Any forest consists of a collection of trees.

### 11. Define a binary tree

A binary tree is a finite set of nodes which is either empty or consists of a root and two disjoint binary trees called the left sub-tree and right sub-tree.

### 12. Define a path in a tree

A path in a tree is a sequence of distinct nodes in which successive nodes are connected by edges in the tree.

### 13. Define a full binary tree

A full binary tree is a tree in which all the leaves are on the same level and every non-leaf node has exactly two children.

### 14. Define a complete binary tree

A complete binary tree is a tree in which every non-leaf node has exactly two children not necessarily to be on the same level.

### 15. State the properties of a binary tree

• The maximum number of nodes on level n of a binary tree is 2n-1, where n≥1.

• The maximum number of nodes in a binary tree of height n is 2n-1, where n≥1.

• For any non-empty tree, nl=nd+1 where nl is the number of leaf nodes and nd is    the number of nodes of degree 2.

10

**16. What is meant by binary tree traversal?**

Traversing a binary tree means moving through all the nodes in the binary tree, visiting each node in the tree only once.

**17. What are the different binary tree traversal techniques?**

• Preorder traversal

• Inorder traversal

• Postorder traversal

• Levelorder traversal

**18. What are the tasks performed during inorder traversal?**

• Traverse the left sub-tree

• Process the root node

• Traverse the right sub-tree

**19. What are the tasks performed during postorder traversal?**

• Traverse the left sub-tree

• Traverse the right sub-tree

• Process the root node

**20. State the merits of linear representation of binary trees.**

• Storage method is easy and can be easily implemented in arrays

• When the location of a parent/child node is known, other one can be determined easily

• It requires static memory allocation so it is easily implemented in all programming language

**21. State the demerit of linear representation of binary trees.**

Insertions and deletions in a node take an excessive amount of processing time due to data movement up and down the array.

**22. State the merit of linked representation of binary trees.**

Insertions and deletions in a node involve no data movement except the rearrangement of pointers, hence less processing time.

**23. State the demerits of linked representation of binary trees.**

• Given a node structure, it is difficult to determine its parent node

• Memory spaces are wasted for storing null pointers for the nodes, which have one or no sub-trees

• It requires dynamic memory allocation, which is not possible in some programming language

**24. Define a binary search tree**

A binary search tree is a special binary tree, which is either empty or it should satisfy the following characteristics:
Every node has a value and no two nodes should have the same value i.e) the values in the binary search tree are distinct

• The values in any left sub-tree is less than the value of its parent node

• The values in any right sub-tree is greater than the value of its parent node

• The left and right sub-trees of each node are again binary search trees

**25. What is the use of threaded binary tree?**

In threaded binary tree, the NULL pointers are replaced by some addresses. The left pointer of the node points to its predecessor and the right pointer of the node points to its successor.

**26. Traverse the given tree using Inorder, Preorder and Postorder traversals.**
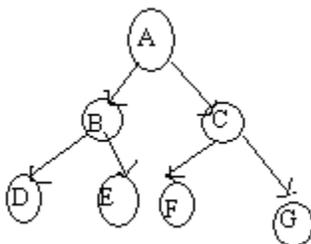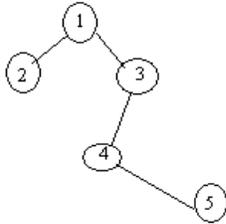


Inorder : D H B E A F C I G J

Preorder: A B D H E C F G I J

Postorder: H D E B F I J G C A

**27. In the given binary tree, using array you can store the node 4 at which location?**



At location 6

| 1 | 2 | 3 | - | - | 4 | - | - | 5 |
|---|---|---|---|---|---|---|---|---|

where LCn means Left Child of node n and RCn means Right Child of node n

**28. Define AVL Tree.**

An empty tree is height balanced. If T is a non-empty binary tree with TL and TR as its left and right subtrees, then T is height balanced if

i) TL and TR are height balanced and

ii) $|hL - hR| \leq 1$

Where hL and hR are the heights of TL and TR respectively.

**29. What do you mean by balanced trees?**

Balanced trees have the structure of binary trees and obey binary search tree properties. Apart from these properties, they have some special constraints, which differ from one data structure to another. However, these constraints are aimed only at reducing the height of the tree, because this factor determines the time complexity.

Eg: AVL trees, Splay trees.

**30.What are the categories of AVL rotations?**

Let A be the nearest ancestor of the newly inserted nod which has the balancing factor ±2. Then the rotations can be classified into the following four categories:

Left-Left: The newly inserted node is in the left subtree of the left child of A.

Right-Right: The newly inserted node is in the right subtree of the right child of

A. Left-Right: The newly inserted node is in the right subtree of the left child of A.

13

Right-Left: The newly inserted node is in the left subtree of the right child of A.

### 31.What do you mean by balance factor of a node in AVL tree?

The height of left subtree minus height of right subtree is called balance factor of a node in AVL tree.The balance factor may be either 0 or +1 or -1.The height of an empty tree is -1.

### 32. Define splay tree.

A splay tree is a binary search tree in which restructuring is done using a scheme called splay. The splay is a heuristic method which moves a given vertex v to the root of the splay tree using a sequence of rotations.

### 33. What is the idea behind splaying?

Splaying reduces the total accessing time if the most frequently accessed node is moved towards the root. It does not require to maintain any information regarding the height or balance factor and hence saves space and simplifies the code to some extent.

# UNIT IV –GRAPHS NON LINEAR DATA STRUCTUURES - GRAPHS

## 1. Define Graph.

A graph G consist of a nonempty set V which is a set of nodes of the graph, a set E which is the set of edges of the graph, and a mapping from the set for edge E to a set of pairs of elements of V. It can also be represented as G=(V, E).

## 2. Define adjacent nodes.

Any two nodes which are connected by an edge in a graph are called adjacent nodes. For example, if an edge x ε E is associated with a pair of nodes (u,v) where u, v ε V, then we say that the edge x connects the nodes u and v.

## 3. What is a directed graph?

A graph in which every edge is directed is called a directed graph.

## 4. What is an undirected graph?

A graph in which every edge is undirected is called a directed graph.

## 5. What is a loop?

An edge of a graph which connects to itself is called a loop or sling.

## 6. What is a simple graph?

A simple graph is a graph, which has not more than one edge between a pair of nodes than such a graph is called a simple graph.

## 7. What is a weighted graph?

A graph in which weights are assigned to every edge is called a weighted graph.

## 8. Define outdegree of a graph?

In a directed graph, for any node v, the number of edges which have v as their initial node is called the out degree of the node v.

### 9. Define indegree of a graph?

In a directed graph, for any node v, the number of edges which have v as their terminal node is called the indegree of the node v.

### 10. Define path in a graph?

The path in a graph is the route taken to reach terminal node from a starting node.

### 11. What is a simple path?

A path in a diagram in which the edges are distinct is called a simple path. It is also called as edge simple.

### 12. What is a cycle or a circuit?

A path which originates and ends in the same node is called a cycle or circuit.

### 13. What is an acyclic graph?

A simple diagram which does not have any cycles is called an acyclic graph.

### 14. What is meant by strongly connected in a graph?

An undirected graph is connected, if there is a path from every vertex to every other vertex. A directed graph with this property is called strongly connected.

### 15. When is a graph said to be weakly connected?

When a directed graph is not strongly connected but the underlying graph is connected, then the graph is said to be weakly connected.

### 16.Namethe        different        ways        of        representing        a        graph?
a.Adjacencymatrix
b. Adjacency list

### 17. What is an undirected acyclic graph?

When every edge in an acyclic graph is undirected, it is called an undirected acyclic graph. It is also called as undirected forest.

**18.   What are the two traversal strategies used in traversing a graph?**

   a.Breadthfirstsearch

   b. Depth first search

**19. What is a minimum spanning tree?**

A minimum spanning tree of an undirected graph G is a tree formed from graph edges that connects all the vertices of G at the lowest total cost.

**20.   Name two algorithms two find minimum spanning tree**

   Kruskal'salgorithm

   Prim's algorithm

**21. Define graph traversals.**

Traversing a graph is an efficient way to visit each vertex and edge exactly once.

**22. List the two important key points of depth first search.**

i) If path exists from one node to another node, walk across the edge – exploring the edge.

ii) If path does not exist from one specific node to any other node, return to the previous node where we have been before – backtracking.

**23. What do you mean by breadth first search (BFS)?**

BFS performs simultaneous explorations starting from a common point and spreading                                out                                independently.

**24.DifferentiateBFSandDFS.**

| No. | DFS | BFS |
|-----|-----|-----|
| 1. | Backtracking is possible from a dead end | Backtracking is not possible |
| 2. | Vertices from which exploration is incomplete are processed in a | The vertices to be explored are organized as a |
| 3. | Search is done in one particular direction | The vertices in the same level are maintained |

### 25. What do you mean by tree edge?

If w is undiscovered at the time vw is explored, then vw is called a tree edge and v becomes the parent of w.

### 26. What do you mean by back edge?

If w is the ancestor of v, then vw is called a back edge.

### 27. Define biconnectivity.

A connected graph G is said to be biconnected, if it remains connected after removal of any one vertex and the edges that are incident upon that vertex. A connected graph is biconnected, if it has no articulation points.

### 28. What do you mean by articulation point?

If a graph is not biconnected, the vertices whose removal would disconnect the graph are known as articulation points.

### 29. What do you mean by shortest path?

A path having minimum weight between two vertices is known as shortest path, in which weight is always a positive number.

### 30. Define Activity node graph.

Activity node graphs represent a set of activities and scheduling constraints. Each node represents an activity (task), and an edge represents the next activity.

### 31. Define adjacency list.

Adjacency list is an array indexed by vertex number containing linked lists. Each node Vi the $i^{th}$ array entry contains a list with information on all edges of G that leave Vi. It is used to represent the graph related problems.

# UNIT V – HASHING AND SET SEARCHING, SORTING AND HASHING TECHNIQUES

1. **Define sorting**

   *Sorting* arranges the numerical and alphabetical data present in a list in a specific order or sequence. There are a number of sorting techniques available. The algorithms can be chosen based on the following factors
   - Size of the data structure
   - Algorithm efficiency
   - Programmer's knowledge of the technique.

2. **Mention the types of sorting**
   - Internal sorting
   - External sorting

3. **What do you mean by internal and external sorting?**

   An **internal sort** is any data sorting process that takes place entirely within the main memory of a computer. This is possible whenever the data to be sorted is small enough to all be held in the main memory.

   **External sorting** is a term for a class of sorting algorithms that can handle massive amounts of data. External sorting is required when the data being sorted do not fit into the main memory of a computing device (usually RAM) and instead they must reside in the slower external memory (usually a hard drive)

4. **Define bubble sort**

   **Bubble sort** is a simple sorting algorithm that works by repeatedly stepping through the list to be sorted, comparing each pair of adjacent items and swapping them if they are in the wrong order. The pass through the list is repeated until no swaps are needed, which indicates that the list is sorted. The algorithm gets its name from the way smaller elements "bubble" to the top of the list.

5. **How the insertion sort is done with the array?**

It sorts a list of elements by inserting each successive element in the previously sorted sublist.

Consider an array to be sorted A[1],A[2],….A[n]

   a. Pass 1 : A[2] is compared with A[1] and placed them in sorted order.
   b. Pass 2 : A[3] is compared with both A[1] and A[2] and inserted at an appropriate place. This makes A[1], A[2],A[3] as a sorted sub array.
   c. Pass n-1 : A[n] is compared with each element in the sub array A[1],A[2],……A[n-1] and inserted at an appropriate position.

6. **What are the steps for selection sort?**

   • The algorithm divides the input list into two parts: the sublist of items already sorted, which is built up from left to right at the front (left) of the list, and the sublist of items remaining to be sorted that occupy the rest of the list.

   • Initially, the sorted sublist is empty and the unsorted sublist is the entire input list.

   • The algorithm proceeds by finding the smallest (or largest, depending on sorting order) element in the unsorted sublist, exchanging it with the leftmost unsorted element (putting it in sorted order), and moving the sublist boundaries one element to the right.

7. **What is meant by shell sort?**

**Shell sort**, also known as **Shell sort** or **Shell's method**, is an in-place comparison sort. It can either be seen as a generalization of sorting by exchange (bubble sort) or sorting by insertion (insertion sort).[1] The method starts by sorting elements far apart from each other and progressively reducing the gap between them. Starting with far apart elements can move some out-of-place elements into position faster than a simple nearest neighbor exchange. Donald Shell published the first version of this sort in 1959. The running time of Shell sort is heavily dependent on the gap sequence it uses

8. **What are the steps in quick sort?**

The steps are:

    a. Pick an element, called a **pivot**, from the list.

    b. Reorder the list so that all elements with values less than the pivot come before the pivot, while all elements with values greater than the pivot come after it (equal values can go either way). After this partitioning, the pivot is in its final position. This is called the **partition** operation.

    c. Recursively apply the above steps to the sub-list of elements with smaller values and separately to the sub-list of elements with greater values.

## 9. Define radix sort

Radix Sort is a clever and intuitive little sorting algorithm. **Radix sort** is a non-comparative integer sorting algorithm that sorts data with integer keys by grouping keys by the individual digits which share the same significant position and value. Radix Sort puts the elements in order by comparing the **digits of the numbers**.

## 10. What are the advantages of insertion sort

Advantages

    a. Simplest sorting technique and easy to implement

    b. It performs well in the case of smaller lists.

    c. It leverages the presence of any existing sort pattern in the list

Disadvantages

- Efficiency of O(n ) is not well suited for large sized lists
- It requires large number of elements to be shifted

## 11. Define searching

Searching refers to determining whether an element is present in a given list of elements or not. If the element is present, the search is considered as successful, otherwise it is considered as an unsuccessful search. The choice of a searching technique is based on the following factors

    a. Order of elements in the list i.e., random or sorted

    b. Size of the list

**12. Mention the types of searching**

The types are

- Linear search
- Binary search

**13. What is meant by linear search?**

**Linear search** or **sequential search** is a method for finding a particular value in a list that consists of checking every one of its elements, one at a time and in sequence, until the desired one is found.

**14. What is binary search?**

For binary search, the array should be arranged in ascending or descending order.
In each step, the algorithm compares the search key value with the middle element of the array. If the key match, then a matching element has been found and its index, or position, is returned.
Otherwise, if the search key is less than the middle element, then the algorithm repeats its action on the sub-array to the left of the middle element or, if the search key is greater, on the sub-array to the right.

**15. Define hashing function**

A hashing function is a key-to-transformation, which acts upon a given key to compute the relative position of the key in an array.
A simple hash function
HASH(KEY_Value)= (KEY_Value)mod(Table-size)

**16. What is open addressing?**

Open addressing is also called closed hashing, which is an alternative to resolve the collisions with linked lists. In this hashing system, if a collision occurs, alternative cells are tired until an empty cell is found.

There are three strategies in open addressing:

- Linear probing
- Quadratic probing
- Double hashing

## 17. What are the collision resolution methods?

The following are the collision resolution methods

- Separate chaining
- Open addressing
- Multiple hashing

## 18. Define separate chaining

It is an open hashing technique. A pointer field is added to each record location, when an overflow occurs, this pointer is set to point to overflow blocks making a linked list. In this method, the table can never overflow, since the linked lists are only extended upon the arrival of new keys

## 19. . Define Hashing.

Hashing is the transformation of string of characters into a usually shorter fixed length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the short hashed key than to find it using the original value.

## 20. What do you mean by hash table?

The hash table data structure is merely an array of some fixed size, containing the keys. A key is a string with an associated value. Each key is mapped into some number in the range 0 to tablesize-1 and placed in the appropriate cell.

## 21. . What do you mean by hash function?

A hash function is a key to address transformation which acts upon a given key to compute the relative position of the key in an array. The choice of hash function should be simple and it must distribute the data evenly. A simple hash function is hash_key=key mod tablesize.

**22. . Write the importance of hashing.**

• Maps key with the corresponding value using hash function.

• Hash tables support the efficient addition of new entries and the time spent on searching for the required data is independent of the number of items stored.

**23. . What do you mean by collision in hashing?**

When an element is inserted, it hashes to the same value as an already inserted element, and then it produces collision.

**24.  What are the collision resolution methods?**

• Separate chaining or External hashing

• Open addressing or Closed hashing

**25. . What do you mean by separate chaining?**

Separate chaining is a collision resolution technique to keep the list of all elements that hash to the same value. This is called separate chaining because each hash table element is a separate chain (linked list). Each linked list contains all the elements whose keys hash to the same index.

**26. . Write the advantage of separate chaining.**

• More number of elements can be inserted as it uses linked lists.

**27. . Write the disadvantages of separate chaining.**

• The elements are evenly distributed. Some elements may have more elements and some may not have anything.

• It requires pointers. This leads to slow the algorithm down a bit because of the time required to allocate new cells, and also essentially requires the implementation of a second data structure.

**28. . What do you mean by open addressing?**

24

Open addressing is a collision resolving strategy in which, if collision occurs alternative cells are tried until an empty cell is found. The cells h0(x), h1(x), h2(x),…. are tried in succession, where hi(x)=(Hash(x)+F(i))mod Tablesize with F(0)=0. The function F is the collision resolution strategy.

**29. What are the types of collision resolution strategies in open addressing?**

      • Linear probing

      • Quadratic probing

      • Double hashing

**30. What do you mean by Probing?**

      Probing is the process of getting next available hash table array cell.

**31. . What do you mean by linear probing?**

Linear probing is an open addressing collision resolution strategy in which F is a linear function of i, F(i)=i. This amounts to trying sequentially in search of an empty cell. If the table is big enough, a free cell can always be found, but the time to do so can get quite large.

**32. What do you mean by primary clustering?**

      In linear probing collision resolution strategy, even if the table is relatively empty, blocks of occupied cells start forming. This effect is known as primary clustering means that any key hashes into the cluster will require several attempts to resolve the collision and then it will add to the cluster.

**33. What do you mean by quadratic probing?**

Quadratic probing is an open addressing collision resolution strategy in which F(i)=i2. There is no guarantee of finding an empty cell once the table gets half full if the table size is not prime. This is because at most half of the table can be used as alternative locations to resolve collisions.

**34. What do you mean by secondary clustering?**

      Although quadratic probing eliminates primary clustering, elements that hash to the same position will probe the same alternative cells. This is known as secondary clustering.

### 35. What do you mean by double hashing?

Double hashing is an open addressing collision resolution strategy in which $F(i)=i.hash2(X)$. This formula says that we apply a second hash function to X and probe at a distance hash2(X), 2hash2(X),….,and so on. A function such as hash2(X)=R-(XmodR), with R a prime smaller than

### 36. What do you mean by rehashing?

If the table gets too full, the running time for the operations will start taking too long and inserts might fail for open addressing with quadratic resolution. A solution to this is to build another table that is about twice as big with the associated new hash function and scan down the entire original hash table, computing the new hash value for each element and inserting it in the new table. This entire operation is called rehashing.

### 37. What is the need for extendible hashing?

If either open addressing hashing or separate chaining hashing is used, the major problem is that collisions could cause several blocks to be examined during a Find, even for a well-distributed hash table. Extendible hashing allows a find to be performed in two disk accesses. Insertions also require few disk accesses.

### 38. List the limitations of linear probing.

- Time taken for finding the next available cell is large.
- In linear probing, we come across a problem known as clustering.

### 39. Mention one advantage and disadvantage of using quadratic probing.

**Advantage:** The problem of primary clustering is eliminated.

**Disadvantage**: There is no guarantee of finding an unoccupied cell once the table is nearly half full.