R.M.D ENGINEERING COLLEGE
DEPARTMENT OF INFORMATION TECHNOLOGY

QUESTION BANK

Subject: CS8391-Data Structures **Year/Sem:II/III**

UNIT-I

PART-A

1.What is called as a Data Structure?

A Data Structure defines a way of representing or organizing all data that contains both the data items and their relationship with each other.

2.Differentiate between data type and data structures.

Data Type: Data Type of a variable is a set of values the variable may assume.

Eg. int, char & real

Data Structures: Data structure mean the way of organizing data values with the help of existing data types.

Eg. Array, stack queue, & tree.

3.What are Primary Data Structures?

- Integers
- Floating-point numbers.
- Character Constants

4.What are the types of Secondary Data Structures?

- Static Data structures
- Dynamic Data structures.

5.What are the static data structures?

- Arrays
- Structures

6.What are the types of dynamic data structures?

- Linear Data Structures
- Non-Linear Data Structures.

7.Give examples of Linear and Non-Linear Data Structures.

- Linear Data Structures
  - i) Linked Lists(Singly & Doubly Linked Lists)
  - ii) Circular Linked Lists(Circular-singly&Circular-doubly Linked Lists).
- Non-Linear Data Structures
  - i) Trees
  - ii) Graphs.

8.What do you mean by Abstract Data Types?

Classes encapsulate all the essential properties of the objects that are to be created. Since the classes use the concept of data abstraction, they are known as Abstract Data Types (ADT).

9.List the operation of set ADT?

Union and Find are the two operations on set ADT

10. Give some applications of stack.

- Evaluation of Expressions.
- Expression conversion.
- Balancing of symbols.

Function Calls.

11. Define Dequeue.

A dequeue is an ordered list in which additions and deletions may be carried out at either end.

12.State the difference between cursor and pointers.

Pointer: Pointer is a variable, which stores the address of another variable.
Using pointers, memory can be allocated dynamically.
Cursor: Cursor is a temporary memory space. Memory allocated is static.

13.What is an ordered list

An ordered list is a linear list which is been ordered by some key.

Eg. An alphabetized list of students in a class, a list of exam scores in decreasing order.

14..State the difference between array and linked list.

Array	Linked List
Contiguous memory location	Memory location need not be necessarily contiguous
Operations such as insertion and deletion are ineffective since the memory locations need to be moved up and down respectively.	Insertion and deletions are easier and needs only one pointer assignment.
Inefficient use of memory space.	A small amount of memory is been wasted for storing a pointer, which is been associated with each node.

15.What are the advantages and disadvantages of linked list?

Advantages:

a. Memory location for a linked list is dynamic, that is memory allocation is done during run time. So memory will not be wasted.

b. Data movements during insertion and deletion will be eliminated. So time will not be wasted.

c. Memory allocation for each node of a linked list need not be continuous.

Disadvantages:

d. Nodes of a linked list cannot be accessed directly. To access a particular node accessing should start only from the beginning.

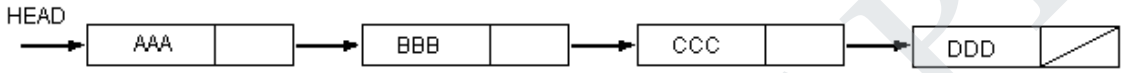
e. To store a single data, along with data, memory must be allocated for a pointer also, which wastes memory.

16.Write the steps required to evaluate the postfix expression.

- Repeatedly read characters from postfix expression
- If the character read is an operand, push the value associated with it into the stack
- If it is an operator, pop the top two values from stack, apply the operator to them and push the result back onto the stack.

17.Define Singly Linked List.

A singly linked list is a list structure in which each node contains a single pointer field that points to the next node in the list, along with a data field.



18.Define Doubly Linked List.

A doubly linked list is a list structure in which each node contains two pointer fields along with a data field namely,

BLINK – Points to the previous node in the list

FLINK – Points to the successive node in the list

PART-B

1. Design a routine to delete an element in a linked list.
2. Develop an algorithm for insertion operation in a singly linked list.
3. Describe in detail about Polynomial manipulation in linked list.
4. What is a linked list? Describe the suitable routine segments for any four operations.
5. Examine the algorithms to implement the doubly linked list and perform all the operations on the created list.
6. Identify the array implementation of list and show all its operation.
7. Discuss the creation of a doubly linked list and appending the list.Give relevant coding in C.
8. Write C code for singly linked list with insert, delete, display operations using structure pointer.
9. Differentiate single linked list and doubly linked list with an example.
10. Explain the application of linked list in detail.
11. Consider an array A[1: n] Given a position, write an algorithm to insert an element in the Array.If the position is empty, the element is inserted easily. If the position is already occupied the element should be inserted with the minimum number of shifts.(Note: The elementscan shift to the left or to the right to make the minimum number of moves).

12. Analyze and write C code for circular linked list with create, insert,delete, display operations.
13. Explain the various operations of the list ADT with examples.
14. Analyze the doubly linked list and circular linked list. Mention its advantages and disadvantages.
15. Explain the steps involved in insertion and deletion into a singly linked list.
16. Recommend an algorithm to add two polynomials when the polynomials are represented singly linked lists.
17. Compose an algorithm toReverse the elements of a single linked lists,count the number of nodes in a given singly linked list. Searching the element from linked list.
18. Given an list 10,20,30,40 ,generalizethe steps to delete a node from the beginning of the linked list, deletion of last node in adeletion of middle node in a list.
19. Develop a C program to split a linked list into sub lists containing odd and even ordered elements in them respectively.

UNIT- 2

PART-A

1. Define stack?

Stack is an ordered collection of items into which new items may be inserted and from which items may be deleted at one end, called the top of stack.

2. Define Circular Queue.

In array based implementation of queue, inefficiency arises only when the value of the rear pointer exceeds maximum size during insertion. This can be rectified if we logically consider the queue to be circular.

In circular queue, once the rear pointer reaches the maximum size, the next location I the first one (index 0), provided the first location is vacant. The specific location is calculated by mod (%) operator.

3.Convert the infix (a+b)*(c+d)/f into postfix & prefix expression

Postfix : a b + c d + * f /
Prefix : / * + a b + c d f

4. Distinguish between stack and queue.

STACK	QUEUE
Insertion and deletion are made at one end.	Insertion at one end rear and deletion at other end front.
The element inserted last would be removed first. So LIFO structure.	The element inserted first would be removed first. So FIFO structure.
Full stack condition: If(top==Maxsize) Physically and Logically full stack	Full stack condition: If(rear = = Maxsize) Logically full. Physically may or may not be full.

5. Define Dequeue.

A dequeue is an ordered list in which additions and deletions may be carried out at either end.

6.What is top of stack?

The end of stack, at which the items are added or deleted, is called top of stack.

7.Why stack is called LIFO list?

T he last element inserted into the stack is the first element to be deleted. For this reason, stack is called Last-In First-Out list.

What are the operations that can be performed on a stack?

Push- Adding an item to a stack. Push(S, i) adds an item i to the top of stack.

Pop - Removes an item from stack. i=Pop(S) returns the top element to i.

8.What is empty stack?

The stack which contains no items in it is called Empty stack.

9.What causes underflow of stack? How it could be avoided?

If an illegal attempt is made to pop or access an item from empty stack, it causes stack underflow.

To avoid underflow, before any pop(S) or stacktop(S), check whether empty(S) is true or false.

10.What are the limitations in the stack, when array used as home of stack?

The Array is finite collection of elements and in stack the number of elements is unlimited. As the stack dynamically changes the attempts to insert more elements than the array size cause overflow

11.Define Modularization?

Modularization is the concept of isolating the complex implementation into set of independent and easily identifiable units, to make the program easily understandable and modifiable.

12. What are the error conditions that could occur in stack implementation? How could they be rectified?

Overflow

Underflow

To avoid overflow, the stack should be checked whether it is full or not before every push operation.

To avoid underflow, the stack should be checked for emptiness before every pop operation.

13. What is overflow in stack?

When array is used as home of stack, and when the stack contains as many elements as array, and as attempt is to push another element on stack, it cause overflow.

14.Why infix expression should be converted to Prefix / Postfix?

In Infix, operator precedence can be set only after scanning the expression for several times. But, conversion of Infix to Postfix or Prefix with use of stack, help to set precedence based on order of arrangement in a single scanning

15. Convert ((A+B) * C – (D – E)) $ (F + G) To Postfix and Prefix notation?

Prefix: $ - * + ABC – DE + FG.

Postfix: AB + C * DE -- FG + $

16.Define Queue?

A queue is an ordered collection of items from which items may be inserted at one end called rear end and into which items may be deleted at the other end called the front end.

17. Why queue is called FIFO list?

In a queue, the first element inserted at rear end will be the first element to be deleted at front end. So, queue is called First-in First-out list.

18.What are the basic operations that could be performed on a Queue?
Insertion – insert (q, x) inserts x at the rear end.
Removal – x=remove (q) remove the element in front end.
Empty(q) – Checks whether queue has any elements or not.

19.What are the limitations of linear queue? How they can be rectified?

When an element is removed from linear queue, the location remains unused. Even if the queue is empty, new elements cannot be inserted. To avoid this, consider queue as a circle, having the first element immediately following the last element.

20. Define Priority Queue?

Priority queue is a data structure in which the intrinsic ordering of the elements determines the results of the basic operations like insertion and removal.

21.What are the two types of priority queues? Briefly explain?
Two types of priority queues are,

i) Ascending priority queue – In this queue, the items can be inserted arbitrarily and only the smallest item will be removed.

ii) Descending priority queue- This allows insertion of items arbitrarily, and only the maximum element from queue will be removed first.

22.What is the difference between queue and priority queue?

In queue the elements can be inserted only at rear end and can be removed only at front end. But in priority queue, elements can be arbitrarily inserted. But the complete queue is searched for removal of high priority element.

23. What are the limitations in priority queue? How it could be rectified?

Deletion of elements makes a location empty. The search of items includes the empty spaces too. This takes more time. To avoid this, use an empty indicator or shift elements forward when each element is deleted. We can also maintain priority queue as an array of ordered elements, to avoid risk in searching.

24..What are the application of Stack?

Two applications
Recursion
Expression

PART-B

1. Describe about stack ADT using array in detail.
2. i)Give an algorithm for push and pop operations on stack using a linked list with an example.
3. ii)Describe the function to examine whether the stack is full() or empty().
4. i)Discuss the process of evaluating postfix expression with an example.
5. ii) Write an algorithm that checks if expression is correctly parenthesized using stack.
6. Give an algorithm to convert an infix expression to a postfix expression using stack with suitable example.

a. Trace the algorithm to convert the infix expression "3-(4/2) +(1*5) + 6" to a postfix expression using stack.
b. Show the simulation using stack for the following expression toconvert infix to postfix :p*q+ (r-s/t).

8. Explain how to evaluate the following arithmetic expressions using stacks.

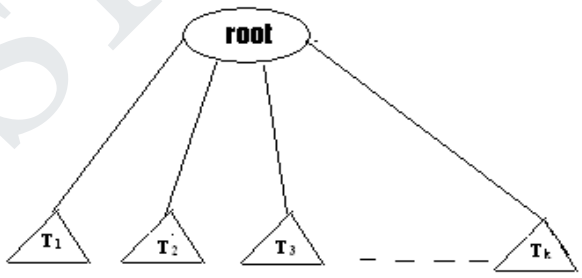
   a. 6 5 2 3 + 8 * + 3 + *2 3 1 * + 9 -

9. Briefly describe the operations of queue with example.
10. Describe about implementation of queue ADT using linked list. Give relevant examples and diagrammatic representations.
11. Discuss and write a C program to implement queue functions using arrays.
12. Explain application of queue with suitable example.
13. Explain circular queue and its implementation.
14. Analyze the implementation of priority queue.
15. Prepare an algorithm to perform the operations in a double ended queue.Develop a C program for linked list implementation of stack.
16. A circular queue has a size of 5 and has 3 elements 10,20 and 40 where F=2 and R=4.After inserting 50 and 60,what is the value of F and R.Trying to insert 30 at this stage what happens? Delete 2 elements from the queue and insert 70, 80 & 90.Assess the sequence of steps with necessary diagrams with the value of F & R.
17. Generalize and develop a function to insert an element into a queue and delete an element from a queue, in which the queue is implemented as a linked list.

UNIT-III
PART-A

1. Define Tree.

A Tree is a collection of one or more nodes with a distinct node called the root , while remaining nodes are partitioned as T_1 ,T_2, ..,T_k , $K \geq 0$ each of which are sub trees, the edges of T_1,T_2,…,T_k are connected the root.



2. Give some applications of Trees.

- Implementing the file system of several operating systems.
- Evaluation of arithmetic expression.
- Set representation.

ecisio n making problems.

3. Define node, degree, siblings, depth/height, level.

Node: A node is an item of information with branches to other items.
Degree: The number of subtrees of a node is called is degree.
Siblings: The children of the same parent is said to be siblings.
Level: The level of a node is defined recursively by assuming the level of the root to be one and if a node is at level l, then its children at level l+1.
Depth/Height: The depth/height of a tree is defined to be the level of a node which is maximum.

4.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.

5.Define terminal nodes in a tree.

A node which has no children is called a terminal node.It is also referred as a leaf node.these nodes have a degree as zero.

6. Define nonterminal nodes in a tree

All intermediate nodes that traverse the given tree from its root node to the terminal nodes are referred as terminal nodes.

7.Define a Binary Tree.

A Binary Tree is a tree,which has nodes either empty or not more than two child nodes,each of which may be a leaf node.

8.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.

9.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.

10.Define a right-skewed binary tree.

A right-skewed binary tree is a tree,which has only right child nodes.

11.State the properties of a Binary Tree.

- Maximum No. of nodes on level n of a binary tree is 2^(n-1),where n>=1.
- Maximum No. of nodes in a Binary tree of height is 2^(n-1),where n>=1.
- For any non-empty tree,nl=nd+1 where nl is the number of leaf nodes and nd is the no. of nodes of degree 2.

12.What are the different ways of representing a Binary Tree?

- Linear Representation using Arrays.
- Linked Representation using Pointers.

13.State the merits of linear representation of binary trees.

- Store methods is easy and can be easily implemented in arrays.
- When the location of the parent/child node is known,other one can be determined easily.
- It requires static memory allocation so it is easily implemented in all programming languages.

ANT AGES of linear representation of binary trees.

- Processing consumes excess of time.
- Slow data movements up and down the array.

15. Define Traversal.

Traversal is an operation which can be performed on a binary tree is visiting all the nodes exactly once.

Inorder: traversing the LST, visiting the root and finally traversing the RST.
Preorder: visiting root, traversing LST and finally traversing RST.
Post- order: traversing LST, then RST and finally visiting root.

16.What are the tasks performed while traversing a binary tree?

- Visiting a node
- Traverse the left structure
- Traverse the right structure.

17.What are the tasks performed during preorder traversal?

- Process the root node
- Traverse the left subtree
- Traverse the right subtree.

Ex : +AB

18.What are the tasks performed during inorder traversal?

- Traverse the left subtree
- Process the root node
- Traverse the right subtree.

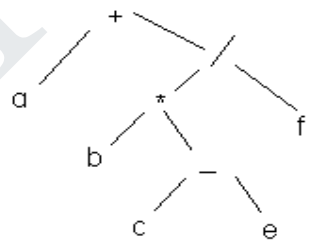
Ex : A+B

19.What are the tasks performed during postorder traversal?

- Traverse the left subtree
- Traverse the right subtree.
- Process the root node.

Ex : AB+

20. Give the pre & postfix form of the expression (a + ((b*(c-e))/f).



21.Define a Binary Search Tree.

A Binary Search Tree is a special binary tree,which is either empty or if it is empty it should satisfy the conditions given below:

- Every node has a value and no two nodes should have the same value(Values should be distinct).
- The value in any left subtree is less than the value of its parent node.
- The value in any right subtree is greater than the value of its parent node.

22.What do u mean by General trees?

General Tree is a tree with nodes having any number of children.

23. Define Forest.

A forest is a collection on N(N>0) disjoint tree or group of trees are called forest. If the root is removed from the tree that tree becomes a forest.

24.Define balanced search tree.

Balanced search tree have the structure of binary tree and obey binary search tree properties with that it always maintains the height as O(log n) by means of a special kind of rotations. Eg. AVL, Splay, B-tree.

25. Define AVL tree.

An empty tree is height balanced. If T is a non-empty binary tree with T_L and T_R as Its left and right subtrees, then T is height balanced if

1. T_L and T_R are height balanced.
2. $| h_L - h_R | \leq 1$.

Where hl and hr are the height of T_L and T_R respectively.

26.What are the drawbacks of AVL trees?

The drawbacks of AVL trees are

- Frequent rotations
- The need to maintain balances for the tree' s nodes
- Overall complexity, especially of the deletion operation.

27. What is a heap?

A heap is a partially ordered data structure, and can be defined as a binary tree assigned to its nodes, one key per node, provided the following two conditions are met

- The tree's shape requirement-The binary tree is essentially complete, that is all the leaves are full except possibly the last level, where only some rightmost leaves will be missing.
- The parental dominance requirement-The key at each node is greater that or equal to the keys of its children

28.What is the main use of heap?

Heaps are especially suitable for implementing priority queues. Priority queue is a set of items with orderable characteristic called an item's priority, with the following operations

- Finding an item with the highest priority
- Deleting an item with highest priority
- Adding a new item to the set

29.Give three properties of heaps?

The properties of heap are

- There exists exactly one essentially complete binary tree with 'n' nodes. Its height is equal to $\log_2 n$
- The root of the heap is always the largest element
- A node of a heap considered with all its descendants is also a heap

30.Give the main property of a heap that is implemented as an array.

A heap can be implemented as an array by recording its elements in the top-down, left-to-right fashion. It is convenient to store the heap's elements in positions 1 through n of such an array. In such a representation

- The parental node keys will be in the first n/2 positions of the array, while the leaf keys will occupy the last n/2 positions
- The children of a key in the array's parental position 'i' ($1 \le i \le n/2$) will be in positions 2i and 2i+1and correspondingly, the parent of the key in position 'i' ($2 \le i \le n$) will be in position i/2.

31.What are the two alternatives that are used to construct a heap?

The two alternatives to construct a heap are

- Bottom-up heap construction
- Top-down heap construction

32.Give the pseudocode for Bottom-up heap construction.

```
ALGORITHM HeapBottomUp(H[1..n])
//Constructs a heap from the elements of the given array
//Input An array H[1..n] of orderable elements
//Output A heap H[1..n]
for I ← n/2 downto 1 do
        k ← I ; v ← H[k]
        heap ← false
        while not heap and 2*k ≤ n do
                j ← 2*k
                if j < n
                        if H[j] < H[j+1] j ←j+1
                if v ≥ H[j]
                        heap ← true
                else H[k] ← H[j]; k ← j
        H[k] ← v
```

33.What is the algorithm to delete the root's key from the heap?

ALGORITHM

- Exchange the root's key with the last key K of the heap
- Decrease the heap's size by one
- "Heapify" the smaller tree by sifting K down the tree exactly in the same way as bottom-up heap construction. Verify the parental dominance for K: if it holds stop the process, if not swap K with the larger of its children and repeat this operation until the parental dominance holds for K in its new position.

34.Who discovered heapsort and how does it work?

Heapsort was discovered by J.W.J. Williams. This is a two stage process that works as follows

- Stage 1 Heap construction: construct a heap for a given array.
- Stage 2 Maximum deletions: Apply the root deletion operation n-1 times to the remaining heap

35.What is a min-heap?

A min-heap is a mirror image of the heap structure. It is a complete binary tree in which every element is less than or equal to its children. So the root of the min-heap contains the smallest element.

A B-tree of order m in an m-way search tree that is either empty or is of height ≥1 and
1. The root node has at least 2 children
2. All nodes other than the root node and failure nodes have at least m/2 children.
3. All failure nodes are at same level.

38. Define Binary Heap?

A Binary Heap is a complete binary tree in which the key value of any node must be lesser than its children is called min heap. If the key value of any node is greater than its children is called max heap.Binary heap is also called as partially ordered tree.

39. Explain array implementation of Binary Heap.

For any element in the array position 'i', the left child is at the position '2i', the right child is at the position '2i+1' and parent is at the position 'i/2'.

40. Define Max-heap.

Maxheap: A heap in which the parent has a larger key that the child's key values then it is called Maxheap.

41. Explain AVL rotation.

Manipulation of tree pointers is centered at the pivot node to bring the tree back into height balance. The visual effect of this pointer manipulation so to rotate the sub tree whose root is the pivot node. This operation is referred as AVL rotation.

42.What are the different type of Rotation in AVL Tree?

Two types of rotation are

1.single rotation

2.double rotation.

PART-B

1. Write an algorithm for preorder, inorder and postorder traversal of a binary tree.
2. Explain the following operations on a binary search tree with suitable algorithms
   a. Find a node
   b. Find the minimum and maximum elements of binary search tree.
3. Write an algorithm for inserting and deleting a node in a binary search tree.
4. Describe the concept of threaded binary tree with example.
5. Discuss in detail the various methods in which a binary tree can be represented. Discuss theadvantage and disadvantage of each method.
6. Consider the following list of numbers14,15,4,9,7,18,3,5,16,4,20,17,9,14,5.Using that construct abinary search tree.
7. Construct B Tree of order m=5 for the following keys 1,12,8,2,25,5,14,28,17,7,52,16,48,68,3,26,29,53,55,45.Delete the keys 8 and 55.State the rules for deletion.
8. Discuss how to insert an element in a AVL tree and explain with algorithm.
9. Explain how deletion can take place in AVL trees with suitable algorithm.
10. What are AVL trees? Describe the different rotations defined for AVL tree.
11. Analyze the operations of B-tree using 2-3 tree with example

13.Explain the construction of expression tree with example. Give the applications of trees

14.Illustrate the construction of binomial heaps and its operations with a suitable example.

15.Illustrate how the delete operation is performed on binary heap?

16.Write suitable operations for percolate up and percolate down operations in a binary heap.

UNIT - 4

PART-A

1. Define Graph.

A Graph G, consists of a set of vertices V, and a set of edges E.V is a finite non-empty set consisting of vertices of vertices of the graph. The set of edges E consists of a pair of vertices from the vertex set.

2.What is undirected graph.

If an edge between any two nodes in a graph is not directionally oriented, a graph is called as undirected graph. It is also called as unqualified graph.

3.What is directed graph.

If an edge between any two nodes in a graph is directionally oriented, a graph is called as directed graph.It is also called as digraph.

4.Define a cycle in a graph.

A cycle is a path containing atleast thee vertices such that the starting and the ending vertices are the same.

5.Define a weakly connected graph.

A directed graph is said to be a weakly connected graph if any vertex doesnt have a directed path to any other vertices.

6.Define a weighted graph.

A graph is said to be weighted graph if every edge in the graph is assigned some weight or value.The weight of an edge is a positive value that may be representing the distance between the vertices or the weights of the edges along the path.

2

5 11

7. Define parallel edges

In some directed as well as undirected graph certain pair of nodes are joined by more than one

alled parallel edges.

8.List some representation of Graphs?

Physically a graph can be represented as,

-adjacency matrix
-Incident matrix
-Adjacency list
-Adjacency multilist
-Circular adjacency list

8.Define Adjacency Matrix.

Adjacency Matrix is a representation used to represent a graph with zeros and ones.A graph containing n vertices can be represented using n rows and n columns.

9.What is meant by Traversing a Graph?

It means visiting all the nodes in the graph

10. Define undirected graph / directed graph.

If G=(V, E) is a graph. The edge between $v_1$and $v_2$is represented as (v_1, v_2). If the edges of the form (v_1, v_2) and (v_2, v_1) are treated as the same edge, then G is said to be an undirected graph.

In case of a directed graph, the edge <v_1, v_2>and <v_2, v_1>are different.

11. Define out degree of a graph.

In a directed graph, for any node v, the number of outgoing edges from vare called out degree of a node v. Ex : out degree of c =2

12. Define In degree of a graph.

In a directed graph, for any node v, the number of incoming edges to vare called In degree of a node v. Ex : In degree of c =1

13. Define total degree of a graph.

The sum of the In degree and out degree of a node is called the total degree of the node. Ex : total degree of a node c = 1 +2 =3

14. Define a path in a graph.

A path in a graph is defined as a sequence of distinct vertices each adjacent to the next,except possibly the first vertex and last vertex is different.

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

P1 = ((a,b),(b,e))
P2 = ((a,c),(c,d),(d,e))

15. What is a complete Graph.

A complete graph is a graph in which there is an edge between every pair of vertices.

16. Give the adjacency list representation for the following

A	B	C	D
0	1	1	1
0	0	0	1
0	0	0	1
1	1	1	0

17. List out the graph traversals of graph search?

The two methods of traversal is,

-Depth First Search (DFS)

-Breadth First Search (BFS)

18. Define minimum cost spanning tree?

A spanning tree of a connected graph G, is a tree consisting of edges and all the vertices of G.

In minimum spanning tree T, for a given graph G, the total weights of the edges of the spanning tree must be minimum compared to all other spanning trees generated from G.

-Prim's and Kruskal is the algorithm for finding Minimum Cost Spanning Tree.

19. Define Shortest path problem?

For a given graph G=(V, E), with weights assigned to the edges of G, we have to find the shortest path (path length is defined as sum of the weights of the edges) from any given source vertex to all the remaining vertices of G.

20.Define topological sort?

A topological sort is an ordering of vertices in a directed acyclic graph, such that if there is a path from v_i to v_j appears after v_i in the ordering.

21.What is the use of Kruskal's algorithm and who discovered it?

Kruskal's algorithm is one of the greedy techniques to solve the minimum spanning tree problem. It was discovered by Joseph Kruskal when he was a second-year graduate student.

22.What is the use of Dijksra's algorithm?

Dijkstra's algorithm is used to solve the single-source shortest-paths problem: for a given vertex called the source in a weighted connected graph, find the shortest path to all its other vertices. The single-source shortest-paths problem asks for a family of paths, each leading from the source to a different vertex in the graph, though some paths may have edges in common.

23. Prove that the maximum number of edges that a graph with n Vertices is n*(n-1)/2.

Choose a vertex and draw edges from this vertex to the remaining n-1 vertices. Then, from these n-1 vertices, choose a vertex and draw edges to the rest of the n-2 Vertices. Continue this process till it ends with a single Vertex.

Hence, the total number of edges added in graph is

(n-1)+(n-2)+(n-3)+…+1 =n*(n-1)/2.

24. Define connected and strongly connected graph.

Two Vertices u and v are said to be connected if there exists a path from u to v in the graph. A directed graph is said to be connected if every pair of vertices in the graph is connected.

A directed graph is said to be strongly connected if for every pair of distinct vertices v_i and v_j, there exists two disjoint paths, one from v_i to v_j and the other from v_j to v_i.

PART-B

1. Examine topological sorting of a graph G with suitable example.
2. Differentiate depth-first search and breadth-first search traversal of a graph with suitable examples.
3. Explain with algorithm, How DFS be performed on a undirected graph.

4. Show the algorithm for finding connected components of an undirected graph using DFS, and derive the time complexity of the algorithm.
5. Discuss an algorithm for Breadth first Search on a graph.
6. Discuss any two applications of Graph with example.
7. Explain the depth first approach of finding articulation points in a connected graph with necessary algorithm.
8. Write short notes on Bi-connectivity.
9. Discuss how to find Euler circuit with an example.

UNIT-V
PART-A

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.

ay to be sorted A[1],A[2],….A[n]

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?**

   x 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.
   x Initially, the sorted sublist is empty and the unsorted sublist is the entire input list.
   x 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. What are the use of hash table?

1.Compilers can use hash table to keep track of declared variable in source code.

2.A hash table is useful for any graph theory problem where nodes haver real names instead of numbers

3.A third use of hash table is in program that play games.

4.On line spell checkers

20, Explain Hashing .

Hashing is a technique used to identify the location of an identifier ‘x’ in the memory by some arithmetic functions like f(x), which gives address of ‘x’ in the table.

21.Explain Hash Function.

Hash Function takes an identifier and computes the address of that identifier in the hash table.

22.Mention Different types of popular hash function.

1.Division method
2.Square method
3.Folding method

23.Define Collision.

When two different keys compute in the same location or address in the hash table through any one of the hashing function then it is termed as collision.

24.Mention Different types of collision resolving techniques.

The collision resolving techniques are:

1.Separate chaining.

2.Open Addressing

Linear Probing

Quadratic Probing

Double Hashing.

PART-B

1. Describe about selection sort with suitable example.
2. Examine the algorithm for Insertion sort and sort the following array: 77, 33, 44, 11, 88, 22, 66, 55
3. List the different types of hashing techniques? Explain them in detail with example.
4. Show the result of inserting the keys 2, 3, 5, 7, 11, 13, 15, 6, 4 into an initially empty extendible hashing data structure with M = 3.
5. Write a C program to search a number with the given set of numbers using binary search.
6. Interpret an algorithm to sort a set of 'N' numbers using bubble sort and demonstrate the sorting steps for the following set of numbers: 88,11,22,44,66,99,32,67,54,10.
7. Discuss the various open addressing techniques in hashing with an example.
8. Sort the given integers and Show the intermediate results using shellsort:35,12,14,9,15,45,32,95,40,5.
9. Write an algorithm to sort an integer array using shell sort.
10. Illustrate with example the open addressing and chaining methods of techniques collision resolution techniques in hashing.
11. Compare working of binary search and linear search technique with example.
12. Analyze extendible hashing in brief.
13. Explain in detail about separate chaining.
14. Formulate the rehashing technique with suitable example.
15. Prepare an algorithm to sortthe elements using radix sort example.
16. Mention the different Sorting methods and Explain about method in detailed Manner.
17. Sort the sequence 96, 31, 27,42,76,61,10,4 using shell sort and radix sort and prepare the required steps.
18. Given input {4371,1323,6173,4199,4344,9679,1989} and a hash function h(x) =x mod10 Prepare the resulting for the following:
    a. Separate chaining hash table.
    b. ii)Open addressing hash table using linear probing.
    c. Open addressing hah table using quadratic probing.
    d. Open addressing hash table with second hashh2(x)=7- (x mod 7).
19. Write and explain non-recursive algorithm for binary search.

20. Using binary search, search the number 26 from the list of numbers and give the steps. 10,7,17,26,32,92