DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE
(AUTONOMOUS)
(Approved by AICTE & Affiliated to Anna University, Chennai)
Re-Accredited by NAAC with 'A' Grade
Accredited by NBA for AERO, BME, CSE, ECE, EEE, IT & MECH.
PERAMBALUR-621212, TAMILNADU, INDIA.
Website: www.dsengg.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

U23ITT32-DATA STRUCTURES QUESTION BANK UNIT I

2 MARKS & 16 MARKS

UNIT I

INTRODUCTION AND LINEAR DATA STRUCTURE – LIST
Introduction to Data structure, Abstract Data Types (ADTs) – List ADT – array-based implementation – linked list implementation —singly linked lists- circularly linked lists- doubly- linked lists – applications of lists – Polynomial ADT – Radix Sort – Multilists

2 MARKS

1. Explain the term data structure.

The data structure can be defined as the collection of elements and all the possible operations which are required for those set of elements. Formally data structure can be defined as a data structure is a set of domains D, a set of domains F and a set of axioms A. this triple (D,F,A) denotes the data structure d.

2. What do you mean by non-linear data structure? Give example.

The non-linear data structure is the kind of data structure in which the data may be arranged in hierarchical fashion. For example- Trees and graphs.

3. What do you linear data structure? Give example.

The linear data structure is the kind of data structure in which the data is linearly arranged. For example- stacks, queues, linked list.

4. Enlist the various operations that can be performed on data structure.

Various operations that can be performed on the data structure are

- Create
- Insertion of element
- Deletion of element
- Searching for the desired element
- Sorting the elements in the data structure
- Reversing the list of elements.

5. What is abstract data type? What are all not concerned in an ADT?

The abstract data type is a triple of D i.e. set of axioms, F-set of functions and A-Axioms in which only what is to be done is mentioned but how is to be done is not mentioned. Thus ADT is not concerned with implementation details.

6. List out the areas in which data structures are applied extensively.

Following are the areas in which data structures are applied extensively.

- Operating system- the data structures like priority queues are used for scheduling the jobs in the operating system.
- Compiler design- the tree data structure is used in parsing the source program. Stack data structure is used in handling recursive calls.
- Database management system- The file data structure is used in database management systems. Sorting and searching techniques can be applied on these data in the file.
- Numerical analysis package- the array is used to perform the numerical analysis on the given set of data.
- Graphics- the array and the linked list are useful in graphics applications.
- Artificial intelligence- the graph and trees are used for the applications like building expression trees, game playing.

7. What is a linked list?

A linked list is a set of nodes where each node has two fields ‘data’ and ‘link’. The data field is used to store actual piece of information and link field is used to store address of next node.

8. What are the pitfall encountered in singly linked list?

Following are the pitfall encountered in singly linked list

- The singly linked list has only forward pointer and no backward link is provided. Hence the traversing of the list is possible only in one direction. Backward traversing is not possible.
- Insertion and deletion operations are less efficient because for inserting the element at desired position the list needs to be traversed. Similarly, traversing of the list is required for locating the element which needs to be deleted.

9. Define doubly linked list.

Doubly linked list is a kind of linked list in which each node has two link fields. One link field stores the address of previous node and the other link field stores the address of the next node.

10. Write down the steps to modify a node in linked lists.

- Enter the position of the node which is to be modified.
- Enter the new value for the node to be modified.
- Search the corresponding node in the linked list.
- Replace the original value of that node by a new value.
- Display the messages as " the node is modified".

11. Difference between arrays and lists.

In arrays any element can be accessed randomly with the help of index of array, whereas in lists any element can be accessed by sequential access only.

Insertion and deletion of data is difficult in arrays on the other hand insertion and deletion of data is easy in lists.

12. State the properties of LIST abstract data type with suitable example.

Various properties of LIST abstract data type are

(i) It is linear data structure in which the elements are arranged adjacent to each other.
(ii) It allows to store single variable polynomial.
(iii) If the LIST is implemented using dynamic memory then it is called linked list. Example of LIST are- stacks, queues, linked list.

13. State the advantages of circular lists over doubly linked list.

In circular list the next pointer of last node points to head node, whereas in doubly linked list each node has two pointers: one previous pointer and another is next pointer. The main advantage of circular list over doubly linked list is that with the help of single pointer field we can access head node quickly. Hence some amount of memory get saved because in circular list only one pointer is reserved.

14. What are the advantages of doubly linked list over singly linked list?

The doubly linked list has two pointer fields. One field is previous link field and another is next link field. Because of these two pointer fields we can access any node efficiently whereas in singly linked list only one pointer field is there which stores forward pointer.

15. Why is the linked list used for polynomial arithmetic?

We can have separate coefficient and exponent fields for representing each term of polynomial. Hence there is no limit for exponent. We can have any number as an exponent.

16. What is the advantage of linked list over arrays?

The linked list makes use of the dynamic memory allocation. Hence the user can allocate or de allocate the memory as per his requirements. On the other hand, the array makes use of the static memory location. Hence there are chances of wastage of the memory or shortage of

memory for allocation.

17. What is the circular linked list?

The circular linked list is a kind of linked list in which the last node is connected to the first node or head node of the linked list.

18. What is the basic purpose of header of the linked list?

The header node is the very first node of the linked list. Sometimes a dummy value such -999 is stored in the data field of header node.

This node is useful for getting the starting address of the linked list.

19. What is the advantage of an ADT?

- **Change:** the implementation of the ADT can be changed without making changes in the client program that uses the ADT.
- **Understandability:** ADT specifies what is to be done and does not specify the implementation details. Hence code becomes easy to understand due to ADT.
- **Reusability:** the ADT can be reused by some program in future.

20. What is static linked list? State any two applications of it.

- The linked list structure which can be represented using arrays is called static linked list.
- It is easy to implement, hence for creation of small databases, it is useful.
- The searching of any record is efficient, hence the applications in which the record need to be searched quickly, the static linked list are used.

21. What is a linked list?

A linked list is a collection of nodes where each node contains two fields: one for storing data and one for storing the address of the next node. It allows dynamic memory allocation and efficient insertion/deletion.

22. What is an Abstract Data Type (ADT)? What is not concerned in an ADT?

An ADT is a specification of a data structure that defines what operations can be performed on it, but not how these operations are implemented. ADTs are not concerned with implementation details.

23. What are the advantages of doubly linked list over singly linked list?

Doubly linked lists allow traversal in both forward and backward directions due to the presence of two pointers in each node, making insertion and deletion more efficient compared to singly linked lists.

24. Why is linked list used for polynomial arithmetic?

Linked lists allow representation of each term of a polynomial with separate fields for coefficient and exponent, offering flexibility with term order and no limit on the exponent values.

25. State any two operations that can be performed on data structures.

Two operations on data structures are:

- Insertion: Adding a new element to the data structure.
- Deletion: Removing an existing element from the data structure.

16 MARKS

1. Explain the insertion operation in linked list. How nodes are inserted after a specified node.

2. Write an algorithm to insert a node at the beginning of list?

3. Discuss the merge operation in circular linked lists.

4. What are the applications of linked list in dynamic storage management?

5. How polynomial expression can be represented using linked list?

6. What are the benefit and limitations of linked list?

7. Define the deletion operation from a linked list.

8. What are the different types of data structure?

9. Explain the operation of traversing linked list. Write the algorithm and give an example.

10. Define the deletion operation from a circularly linked list.

DEPARTMENT OF INFORMATION TECHNOLOGY
U23ITT32 – DATA STRUCTURES
UNIT II
2 MARKS & 16 MARKS

UNIT II LINEAR DATA STRUCTURES – STACKS, QUEUES
Stack ADT – Operations - Applications - Evaluating arithmetic expressions- Conversion of Infix to postfix expression - Queue ADT – Operations - Circular Queue – Priority Queue - deQueue – applications of queues

2MARKS

1. Define Stack

A Stack is an ordered list in which all insertions (Push operation) and deletion (Pop operation) are made at one end, called the top. The topmost element is pointed by top. The top is initialized to -1 when the stack is created that is when the stack is empty. In a stack S = (a1,an), a1 is the bottom most element and element a is on top of element ai-1. Stack is also referred as Last In First Out (LIFO) list.

2. What are the various Operations performed on the Stack?

The various operations that are performed on the stack are

CREATE(S) – Creates S as an empty stack.

PUSH(S,X) – Adds the element X to the top of the stack.

POP(S) – Deletes the top most elements from the stack.

TOP(S) – returns the value of top element from the stack.

ISEMTPTY(S) – returns true if Stack is empty else false.

ISFULL(S) - returns true if Stack is full else false.

3. How do you test for an empty stack?

The condition for testing an empty stack is top =-1, where top is the pointer pointing to the topmost element of the stack, in the array implementation of stack. In linked list implementation of stack the condition for an empty stack is the header node link field is NULL.

4. Name two applications of stack?

Nested and Recursive functions can be implemented using stack. Conversion of Infix to Postfix expression can be implemented using stack. Evaluation of Postfix expression can be implemented using stack.

5. **Define a suffix expression.**

The notation used to write the operator at the end of the operands is called suffix notation.

Suffix notation format : operand operand operator

Example: ab+, where a & b are operands and '+' is addition operator.

6. **What do you meant by fully parenthesized expression? Give example.**

A pair of parentheses has the same parenthetical level as that of the operator to

which it corresponds. Such an expression is called fully parenthesized expression.

Ex: (a+((b*c) + (d * e))

7. **Write the postfix form for the expression -A+B-C+D?**

A-B+C-D+

8. **What are the postfix and prefix forms of the expression?**

A+B*(C-D)/(P-R)

Postfix form: ABCD-*PR-/+

Prefix form: +A/*B-CD-PR

9. **Explain the usage of stack in recursive algorithm implementation?**

In recursive algorithms, stack data structures is used to store the return address when a recursive call is encountered and also to store the values of all the parameters essential to the current state of the function.

10. **Define Queues.**

A Queue is an ordered list in which all insertions take place at one end called the rear, while all deletions take place at the other end called the front. Rear is initialized to -1 and front is initialized to 0. Queue is also referred as First In First Out (FIFO) list.

11. **What are the various operations performed on the Queue?**

The various operations performed on the queue are

CREATE(Q) – Creates Q as an empty Queue.

Enqueue(Q,X) – Adds the element X to the Queue.

Dequeue(Q) – Deletes a element from the Queue.

ISEMTPTY(Q) – returns true if Queue is empty else false.

ISFULL(Q) - returns true if Queue is full else false.

12. **How do you test for an empty Queue?**

The condition for testing an empty queue is rear=front-1. In linked list implementation of queue the condition for an empty queue is the header node link field is NULL.

13. **Write down the function to insert an element into a queue, in which the queue is implemented as an array. (May 10)**

Q – Queue

X – element to added to the queue Q

IsFull(Q) – Checks and true if Queue Q is full

Q->Size - Number of elements in the queue Q

Q->Rear – Points to last element of the queue Q

Q->Array – array used to store queue elements

```
void enqueue (int X, Queue Q) {
        if(IsFull(Q))
          Error ("Full queue");
        else       {
                Q->Size++;
                    Q->Rear = Q->Rear+1;
                    Q->Array[ Q->Rear ]=X;
                }
        }
```

14. Define Dequeue.

Deque stands for Double ended queue. It is a linear list in which insertions and deletion are made from either end of the queue structure.

15. **Define Circular Queue.**

Another representation of a queue, which prevents an excessive use of memory by arranging elements/ nodes $Q_1, Q_2, \ldots Q_n$ in a circular fashion. That is, it is the queue, which wraps around upon reaching the end of the queue\

16. What is the use of a stack in evaluating arithmetic expressions?

Stacks are used to evaluate postfix expressions by storing operands. When an operator is encountered, the top two operands are popped, the operation is performed, and the result is pushed back. This continues until the expression is fully evaluated.

17. Why is stack called a Last In First Out (LIFO) structure?
In a stack, the most recently added element is the first one to be removed. This behavior is referred to as Last In First Out (LIFO), as the insertion and deletion happen only at one end—called the top.

18. What is the role of a stack in infix to postfix conversion?
During conversion, a stack is used to hold operators and parentheses. It ensures that the correct order of operations and precedence rules are followed by temporarily storing and then popping operators as needed.

19. What is a priority queue?
A priority queue is a special type of queue where each element has a priority. Elements with higher priority are served before those with lower priority, regardless of their order in the queue.

20. What is the difference between a linear queue and a circular queue?
In a linear queue, once the rear reaches the end of the array, no more elements can be added even if space is available at the front. A circular queue reuses this space by connecting the rear back to the front, forming a loop.

21. List two applications of queues.
Queues are used in CPU scheduling for managing process execution order and in printer spooling to manage multiple print jobs waiting to be printed in the order they arrived.

22. What is the primary disadvantage of a simple linear queue implemented using arrays?
The major drawback is inefficient memory usage. Once the front elements are dequeued, their space cannot be reused unless the elements are shifted manually, leading to wastage.

23. What is the main benefit of using a circular queue over a linear queue?
A circular queue allows reusing the empty spaces left by dequeued elements. This improves memory utilization and avoids the need to shift elements after deletions.

24. Mention two operations specific to a DeQueue.
In a DeQueue (Double Ended Queue), insertion and deletion can be done at both ends. Operations like `InsertFront` and `DeleteRear` are examples that differentiate it from standard queues.

25. When does overflow occur in a circular queue?
Overflow in a circular queue occurs when the next position of the rear pointer equals the front pointer, meaning there is no free space left to add new elements even though the array is circular.

16 MARKS

1. Write an algorithm for Push and Pop operations on Stack using Linked list. (8)
2. Explain the linked list implementation of stack ADT in detail?
3. Define an efficient representation of two stacks in a given area of memory with n words and explain.
4. Explain linear linked implementation of Stack and Queue?
    a. Write an ADT to implement stack of size N using an array. The elements in the stack are to be integers. The operations to be supported are PUSH, POP and DISPLAY. Take into account the exceptions of stack overflow and stack underflow. (8)

b. 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. Show the sequence of steps with necessary diagrams with the value of F & R. (8 Marks)

5. Write the algorithm for converting infix expression to postfix (polish) expression?
6. Explain in detail about priority queue ADT in detail?
7. Write a function called 'push' that takes two parameters: an integer variable and a stack into which it would push this element and returns a 1 or a 0 to show success of addition or failure.
8. What is a DeQueue? Explain its operation with example?
9. Explain the array implementation of queue ADT in detail?
10. Explain the addition and deletion operations performed on a circular queue with necessary algorithms.(8) (Nov 09)

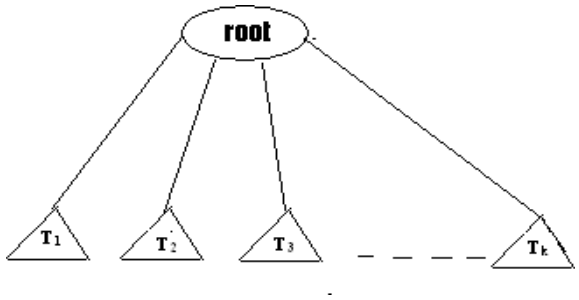
DEPARTMENT OF INFORMATION TECHNOLOGY
DATA STRUCTURES – U23ITT32
UNIT-III

NON-LINEAR DATA STRUCTURES – TREES
Tree ADT – tree traversals - Binary Tree ADT – expression trees – applications of trees – binary search tree ADT – Threaded Binary Trees- AVL Trees – B-Tree - B+ Tree - Heap – Applications of heap.

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 subtrees, the edges of $T_1,T_2,\ldots,T_k$ are connected the root.



2. Give some applications of Trees.

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

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

Node: Anode is an item of information with branches too the r 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 term in a l 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 non terminal nodes in a tree

All in term Mediate 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.

Aright-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 no f 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, n l=nd+1 where n l is the number of leaf nodes and n d 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 allocations to it is easily implemented in all programming languages.
- 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.

In order: traversing the LST, visiting the root and finally traversing the RST.

Pre order: 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 pre order traversal?

- Process the root node
- Traverse the left sub tree
- Traverse the right sub tree. Ex : +AB

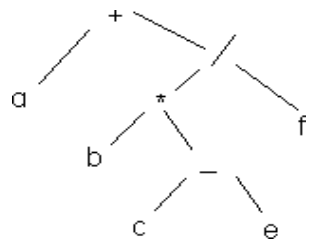
18. What are the tasks performed during in order traversal?

- Traverse the left sub tree
- Process the root node
- Traverse the right sub tree. Ex : A+B

19. What are the tasks performed during post order traversal?

- Traverse the left sub tree
- Traverse the right sub tree.
- Process the root node.
  Ex : AB+

20. Give the pre &post fix for 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 sub tree is less than the value of its parent node.
- The value in any right sub tree 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 sub trees, then T is height balanced if

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

Where h l and hr are the height of T_L and T_R respectively.

PART-B

1. Write an algorithm for preorder, in order and post order 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 binarysearch 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 the advantage 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 a binary search tree.
7. Construct B Tree of order m=5for 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 using2-3 tree with example13. 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.

DEPARTMENT OF INFORMATION TECHNOLOGY
DATA STRUCTURES – U23ITT32
UNIT-IV

NON-LINEAR DATA STRUCTURES - GRAPHS
Graph Definition – Representation of Graphs – Types of Graph - Breadth-first traversal – Depth- first traversal — Bi-connectivity – Euler circuits – Topological Sort – Dijkstra's algorithm – Minimum Spanning Tree – Prim's algorithm – Kruskal's algorithm

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 graphis 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 tobe a weakly connected graph if any vertex doesn't 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.

7. **Define parallel edges**

In some directed as well as undirected graph certain pair of nodes are joined by more than one called 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 are presentation 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 visitingallthenodes in the graph.

10. Define undirected graph/directed graph.

If G=(V,E) is a graph. The edge between v_1 and $v_2$isrepresentedas(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 out going edges from v are 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 v are 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 take n to reach the terminal node from a starting node. The path from a to e are

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. Gi**ve 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-1vertices. 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.

25. List out few of the Application of tree data-structure?

- The manipulation of Arithmetic expression Ø Used for Searching Operation
- Used to implement the file system of several popular operating systems Ø Symbol Table construction
- Syntax analysis

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.
10. Differentiate Prim's and kruskal algorithm with suitable examples.

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE
(AUTONOMOUS)
(Approved by AICTE & Affiliated to Anna University, Chennai)
Re-Accredited by NAAC with 'A' Grade
Accredited by NBA for AERO, BME, CSE, ECE, EEE, IT & MECH.
PERAMBALUR-621212, TAMILNADU, INDIA.
Website: www.dsengg.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY
DATA STRUCTURES – U23ITT32
UNIT-V
SEARCHING, SORTING AND HASHING TECHNIQUES

SEARCHING, SORTING AND HASHING TECHNIQUES
Searching – Linear Search – Binary Search. Sorting – Bubble sort – Selection sort – Insertion sort – Shell sort – Merge Sort – Hashing – Hash Functions – Separate Chaining – Open Addressing – Rehashing – Extendible Hashing

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. **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)

3. **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.

4. How the insertion sort is done with the array?

It sorts a list of elements by inserting each successive element in the previously sorted sub list. way to be sorted A[1],A[2],….A[n]

b. Pass2: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 subarray A[1],A[2],……A[n-1] and inserted at an appropriate position.

5. What are the steps for selection sort?

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

X Initially, the sorted sub list is empty and the unsorted sub list is the entire input list.

X The algorithm proceeds by finding the smallest (or largest, depending on sorting order) element in the unsorted sub list, exchanging it with the left most unsorted element (putting it in sorted order), and moving the sub list boundaries one element to the right.

6. 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 in1959. The running time of Shell sort is heavily dependent on the gap sequence it uses

7. 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.

8. 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**.

9. 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

10. 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

11. Mention the types of searching

The types are

- Linear search
- Binary search

12. 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.

13. 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.

14. 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)

15. 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: x Linear probing x Quadratic probing x

Double hashing

16. What are the collision resolution methods?

The following are the collision resolution methods

- ❖ Separate chaining
- ❖ Open addressing
- ❖ Multiple hashing

17. 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.

18. 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. Online 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.

19. What is the difference between linear search and binary search?

Linear search scans each element sequentially until the target is found or the list ends.
Binary search repeatedly divides the sorted list in half to locate the element.
Binary search is more efficient but requires the input to be sorted.

20. What is bubble sort and how does it work?

Bubble sort works by repeatedly comparing adjacent elements and swapping them if they are in the wrong order.
This continues until the entire list is sorted in the desired order.
Although simple, its worst-case time complexity is $O(n^2)$.

21. What is the key idea behind insertion sort?

Insertion sort builds the sorted list one item at a time by inserting each element into its correct position.
It performs well on small or nearly sorted lists due to fewer comparisons.
Its time complexity is $O(n^2)$ in the worst case and $O(n)$ in the best case.

22. What is the purpose of a hash function in hashing?
A hash function computes an index from a key to determine where to store it in the hash table.
It plays a key role in ensuring quick data retrieval and efficient use of space.
A good hash function distributes keys uniformly and minimizes collisions.

23. What is open addressing in hashing?
Open addressing resolves collisions by finding another open slot in the table through probing.
Common probing methods include linear, quadratic, and double hashing.
All elements are stored within the hash table itself, unlike separate chaining.

24. Define shell sort. How is it different from insertion sort?
Shell sort improves insertion sort by comparing elements separated by a gap, which reduces over time.
This allows distant elements to move closer to their final position quickly.
It performs better than insertion sort, especially for larger lists.

25. What is separate chaining in hashing?
Separate chaining handles collisions by maintaining a linked list for each hash table index.
When multiple keys hash to the same index, they are stored in that list.
It allows dynamic storage and can handle more elements than the table size.

PART-B

1. Describe about selection sort with suitable example.
2. Examine the algorithm for Insertion sort and sort the followingarray:77,33, 44, 11, 88, 22, 66, 55
3. List the different types of hashing techniques? Explain the min detail with example.
4. Show the result ofinsertingthekeys2,3,5,7,11,13,15,6,4into 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 there hashing technique with suitable example.
15. Prepare an algorithm to sort the 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 ha h 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.