ST.JOSEPH COLLEGE OF ENGINEERING
DEPARTMENT OF INFORMATION TECHNOLOGY
QUESTION BANK

CS8391 DATA STRUCTURES

OBJECTIVES:

- To understand the concepts of ADTs
- To Learn linear data structures– lists, stacks, and queues
- To understand sorting, searching and hashing algorithms
- To apply Tree and Graph structures

UNIT I LINEAR DATA STRUCTURES – LIST 9

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 Manipulation – All operations
(Insertion, Deletion, Merge, Traversal).

UNIT II LINEAR DATA STRUCTURES – STACKS, QUEUES 9

Stack ADT – Operations - Applications - Evaluating arithmetic expressions- Conversion of Infix to postfix expression - Queue ADT – Operations - Circular Queue – Priority Queue - deQueue – applications of queues.

UNIT III NON LINEAR DATA STRUCTURES – TREES 9

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.

UNIT IV NON LINEAR DATA STRUCTURES - GRAPHS 9

Definition – Representation of Graph – Types of graph - Breadth-first traversal - Depth-first traversal – Topological Sort – Bi-connectivity – Cut vertex – Euler circuits – Applications of graphs. 5

UNIT V SEARCHING, SORTING AND HASHING TECHNIQUES 9 Searching- Linear Search - Binary Search. Sorting - Bubble sort - Selection sort - Insertion sort - Shell sort – Radix sort. Hashing- Hash Functions – Separate Chaining – Open Addressing – Rehashing – Extendible Hashing.

TOTAL: 45 PERIODS OUTCOMES: At the end of the course, the student should be able to:

- Implement abstract data types for linear data structures.
- Apply the different linear and non-linear data structures to problem solutions.
- Critically analyze the various sorting algorithms.

TEXT BOOKS:

1. Mark Allen Weiss, “Data Structures and Algorithm Analysis in C”, 2nd Edition, Pearson Education,1997.
2. Reema Thareja, “Data Structures Using C”, Second Edition , Oxford University Press, 2011

REFERENCES: 1. Thomas H. Cormen, Charles E. Leiserson, Ronald L.Rivest, Clifford Stein, “Introduction to Algorithms", Second Edition, Mcgraw Hill, 2002. 2. Aho, Hopcroft and Ullman, “Data Structures and Algorithms”, Pearson Education,1983. 3. Stephen G. Kochan, “Programming in C”, 3rd edition, Pearson Education. 4. Ellis Horowitz, Sartaj Sahni, Susan Anderson-Freed, “Fundamentals of Data Structures in C”, Second Edition, University Press, 2008

UNIT- I

PART A

1. **Define: data structure.**
   A data structure is a way of storing and organizing data in the memory for efficient usage. The way information is organized in the memory of a computer.

2. **Give few examples for data structures?**
   Arrays, stacks, queue, list, tree, graph, set, map, table and deque.

3. **What are the different types of data structures?**
   i) Primitive
   ii) Composite
   iii) Abstract

4. **What are primitive data types?**
   The basic building blocks for all data structures are called primitive data types.
   (e.g) int, float, char, double, Boolean

5. **What are composite data types?**
   Composite data types are composed of more than one primitive data type.
   (e.g) array,structure,union

6. **What is meant by an abstract data type?**
   An ADT is a mathematical model for a certain class of data structures that have similar behavior. (e.g) list, stack, queue

7. **How can we categorize data structures based on data access?**
   Linear – list, stack, queue
   Non-linear- heap, tree, graph

8. **State the difference between linear and non-linear data structures.**
   The main difference between linear and nonlinear data structures lie in the way they organize data elements.
   In linear data structures, data elements are organized sequentially and therefore they are easy to implement in the computer's memory.
   In nonlinear data structures, a data element can be attached to several other data elements to represent specific relationships that exist among them. Due to this it might be difficult to be implemented in computer's linear memory.

9. **List a few real-time applications of data structures?**
   - Undo and redo feature - stack
   - Decision making - graph
   - Printer (printing jobs) – queue

- Compilers – hash table
- Directory structure- trees
- Communication networks- graphs

10. Define List.

The general form of the list is a_1, a_2, a_3… a_n. The size of the list is 'n'. Any element in the list at the position i is defined to be at a_i, a_{i+1} the successor of a_i , and a_{i-1} is the predecessor of a_i. a_1 doesn't have predecessor and a_n doesn't have successor.

11. What are the various operations done on List ADT?

The operations done under List ADT are Print list, Insert, Delete, FindPrevious, Find k^{th}, Find, MakeEmpty, IsLast and IsEmpty.

12. What are the different ways to implement list?

- Array implementation of list
- Linked list implementation of list
- Cursor implementation of list

13. Arrays are not used to implement lists. Why?

- Requires that the list size to be known in advance
- Running time for insertions and deletions is slow

14. What are the advantages in the array implementation of list?

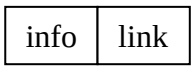
- Print list operation can be carried out at linear time
- Finding K^{th} element takes a constant time

15. What are the disadvantages in the array implementation of list?

The running time for insertions and deletions is so slow and the list size must be known in advance.

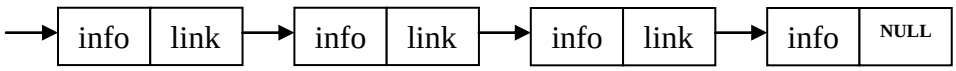
16. Define node.

A node consists of two fields namely an information field called INFO and a pointer field called LINK. The INFO field is used to store the data and the LINK field is used to store the address of the next field.



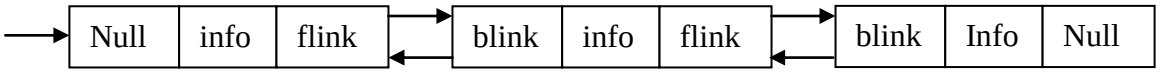
17. What is a linked list?

Linked list is series of nodes, which are not necessarily adjacent in memory. Each node contains a data element and a pointer to the next node.



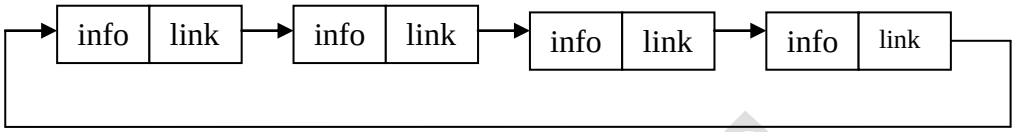
18. What is a doubly linked list?

In a doubly linked list, along with the data field there will be two pointers one pointing the next node(flink) and the other pointing the previous node(blink).



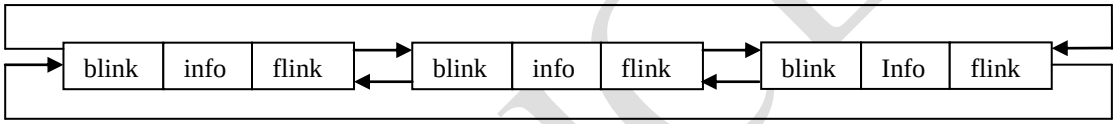
19. Define circularly linked list?

In a singly circular linked list the last node's link points to the first node of the list.



20. Define double circularly linked list?

In a circular doubly linked list the last node's forward link points to the first ode of the list, and the first node's back link points to the last node of the list.



21. Mention the disadvantages of circular list?

The disadvantage of using circular list is

- It is possible to get into an infinite loop.
- It is not possible to detect the end of the list.

22. What is the need for the header?

Header of the linked list is the first element in the list and it may store the number of elements in the list. It points to the first data element of the list.

Without header

- Insertion at the front of the list needs special coding & updating of the linked list address.
- Deletion at the front of the list also needs special coding & updating of the linked list address.

23. List three applications that uses linked list?

Three examples/applications that uses linked list are Polynomial representation, radix sort, & multi lists.

PART-B

1. Derive an ADT to perform insertion and deletion in a singly linked list.(8) (Nov 10)
2. Design an algorithm to reverse the linked list. Trace it with an example?(8)
3. Write an algorithm for inserting and deleting an element from Circular linked list?(8)
4. Write the algorithm for the deletion and reverse operations on doubly linked list. (8) (Nov 09)
5. Write algorithms to perform the following in doubly linked list: (may 10)
   (i) To insert an element in the beginning, middle, end of the list. (8)

(ii) To delete an element from anywhere in the list. (8)
An element is a structure variable that contains an integer data field and a string data field.

6. Write an algorithm to perform the following polynomial manipulation using linked list representation
Insertion, Deletion, Merge, Traversal

UNIT-II

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 ai 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 eg.

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 Deque.

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

Part – B

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. (6)
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)

UNIT- III

1. Define tree?

Trees are non-liner data structure, which is used to store data items in a shorted sequence. It represents any hierarchical relationship between any data Item. It is a collection of nodes, which has a distinguish node called the root and zero or more non-empty sub trees T1, T2,….Tk. each of which are connected by a directed edge from the root.

2. Define Height of tree?

The height of n is the length of the longest path from root to a leaf. Thus all leaves have height zero. The height of a tree is equal to a height of a root.

3. Define Depth of tree?

For any node n, the depth of n is the length of the unique path from the root to node n. Thus for a root the depth is always zero.

4. What is the length of the path in a tree?

The length of the path is the number of edges on the path. In a tree there is exactly one path form the root to each node.

5. Define sibling?

Nodes with the same parent are called siblings.

6. Define binary tree?

A Binary tree is a finite set of data items which is either empty or consists of a single item called root and two disjoin binary trees called left sub tree max degree of any node is two.

7. What are the two methods of binary tree implementation?

Two methods to implement a binary tree are,

a. Linear representation.

b. Linked representation

8. What are the applications of binary tree?

Binary tree is used in data processing.

a. File index schemes

b. Hierarchical database management system

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

Ø The manipulation of Arithmetic expression

Ø Used for Searching Operation

Ø Used to implement the file system of several popular operating systems

Ø Symbol Table construction

Ø Syntax analysis

10. Define expression tree?

Expression tree is also a binary tree in which the leafs terminal nodes or operands and non-terminal intermediate nodes are operators used for traversal.

11. Define tree traversal and mention the type of traversals?

Visiting of each and every node in the tree exactly is called as tree traversal.

Three types of tree traversal

1. Inorder traversal

2. Preoder traversal

3. Postorder traversal.

12. Define in -order traversal?

In-order traversal entails the following steps;

a. Traverse the left subtree

b. Visit the root node

c. Traverse the right subtree

13. Define threaded binary tree.

A binary tree is threaded by making all right child pointers that would normally be null point to the inorder successor of the node, and all left child pointers that would normally be null point to the inorder predecessor of the node.

14. What are the types of threaded binary tree?

Right-in threaded binary tree

Left-in threaded binary tree

Fully-in threaded binary tree

15. Define Binary Search Tree.

Binary search tree is a binary tree in which for every node X in the tree, the values of all the keys in its left subtree are smaller than the key value in X and the values of all the keys in its right subtree are larger than the key value in X.

16.What is AVL Tree?

AVL stands for Adelson-Velskii and Landis.An AVL tree is a binary search tree which has the following properties:

1. The sub-trees of every node differ in height by at most one.
2. Every sub-tree is an AVL tree.

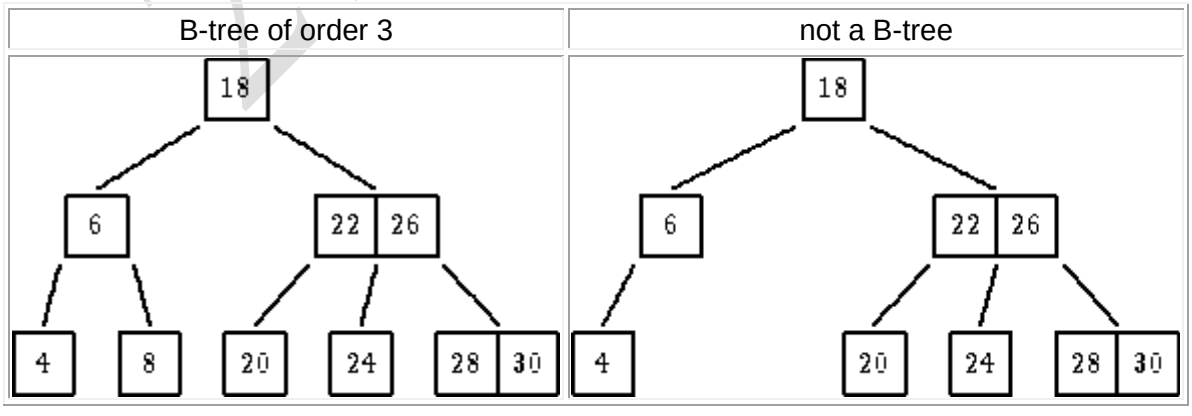
Search time is O(logn). Addition and deletion operations also take O(logn) time.

17. List out the steps involved in deleting a node from a binary search tree.

Deleting a node is a leaf node (ie) No children
Deleting a node with one child.
Deleting a node with two Childs.

18. What is 'B' Tree?

A B-tree is a tree data structure that keeps data sorted and allows searches, insertions, and deletions in logarithmic amortized time. Unlike self-balancing binary search trees, it is optimized for systems that read and write large blocks of data. It is most commonly used in database and file systems.



Important properties of a B-tree:

- B-tree nodes have many more than two children.
- A B-tree node may contain more than just a single element.

19. What is binomial heaps?

A binomial heap is a collection of binomial trees that satisfies the following binomial-heap properties:

1. No two binomial trees in the collection have the same size.
2. Each node in each tree has a key.
3. Each binomial tree in the collection is heap-ordered in the sense that each non-root has a key strictly less than the key of its parent

.

The number of trees in a binomial heap is O(log n).

20. Define complete binary tree.

If all its levels, possible except the last, have maximum number of nodes and if all the nodes in the last level appear as far left as possible.

PART - B

1. Explain the AVL tree insertion and deletion with suitable example.
2. Describe the algorithms used to perform single and double rotation on AVL tree.
3. Explain about B-Tree with suitable example.
4. Explain about B+ trees with suitable algorithm.
5. Write short notes on
    1) Binomial heaps
    2) Fibonacci heaps
6. Explain the tree traversal techniques with an example.
7. Construct an expression tree for the expression (a+b*c) + ((d*e+f)*g). Give the outputs when you apply inorder, preorder and postorder traversals.
8. How to insert and delete an element into a binary search tree and write down the code for the insertion routine with an example.
9. What are threaded binary tree? Write an algorithm for inserting a node in a threaded binary tree.
10. Create a binary search tree for the following numbers start from an empty binary search tree. 45,26,10,60,70,30,40 Delete keys 10,60 and 45 one after the other and show the trees at each stage.

UNIT- IV

PART A

1. Write the definition of weighted graph?

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

2. 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 of edges E to set of pairs of elements of V. It can also be represented as G=(V, E).

3. Define adjacency matrix?

The adjacency matrix is an n x n matrix A whose elements aij are given by
aij = 1 if (vi, vj) Exists =0 otherwise

4. 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□ E is associated with a pair of nodes
(u,v) where u, v □ V, then we say that the edge x connects the nodes u and v.

5. What is a directed graph?

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

6. What is an undirected graph?

A graph in which every edge is undirected is called an undirected graph.

7. What is a loop?

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

8. What is a simple graph?

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

9. What is a weighted graph?

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

10. Define indegree and out degree 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.

Outdegree: Number of edges having the node v as root node is the outdegree of the node v.

11. Define path in a graph?

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

12. 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.

13. What is a cycle or a circuit?

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

14. What is an acyclic graph?

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

15. 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.

16. When 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.

17. Name the different ways of representing a graph? Give examples (Nov 10)

a. Adjacency matrix
b. Adjacency list

18. 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.

19. What is meant by depth?

The depth of a list is the maximum level attributed to any element with in the list or with in any sub list in the list.

20. What is the use of BFS?

BFS can be used to find the shortest distance between some starting node and the remaining nodes of the graph. The shortest distance is the minimum number of edges traversed in order to travel from the start node the specific node being examined.

21. What is topological sort?

It is an ordering of the vertices in a directed acyclic graph, such that: If there is a path from u to v, then v appears after u in the ordering.

22. Write BFS algorithm

1. Initialize the first node's dist number and place in queue
2. Repeat until all nodes have been examined
3. Remove current node to be examined from queue
4. Find all unlabeled nodes adjacent to current node
5. If this is an unvisited node label it and add it to the queue
6. Finished.

23. Define biconnected graph?

A graph is called biconnected if there is no single node whose removal causes the graph to break into two or more pieces. A node whose removal causes the graph to become disconnected is called a cut vertex.

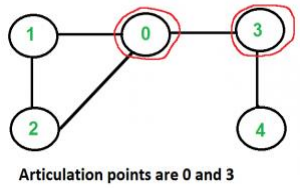
24. What are the two traversal strategies used in traversing a graph?

a. Breadth first search
b. Depth first search

25. **Articulation Points (or Cut Vertices) in a Graph**

A vertex in an undirected connected graph is an articulation point (or cut vertex) iff removing it (and edges through it) disconnects the graph. Articulation points represent vulnerabilities in a connected network – single points whose failure would split the network into 2 or more disconnected components. They are useful for designing reliable networks.

For a disconnected undirected graph, an articulation point is a vertex removing which increases number of connected components.
Following are some example graphs with articulation points encircled with red color.



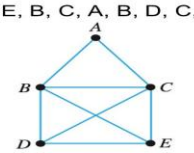
Articulation points are 0 and 3

26. An **Euler** path is a path that uses every edge of a**graph** exactly once. An **Euler circuit** is a **circuit** that uses every edge of a **graph** exactly once. ▸ An **Euler**path starts and ends at different vertices. ▸ An **Euler circuit** starts and ends at the same vertex.

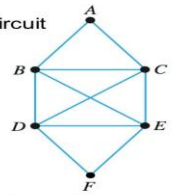
Examples

- Euler path

D, E, B, C, A, B, D, C, E



- Euler circuit



D, E, B, C, A, B, D, C, E, F, D

PART-B

1. Explain the various representation of graph with example in detail?
2. Explain Breadth First Search algorithm with example?
3. Explain Depth first and breadth first traversal?
4. What is topological sort? Write an algorithm to perform topological sort?(8) (Nov 09)
5. (i) write an algorithm to determine the biconnected components in the given graph. (10) (may 10)
   (ii)determine the biconnected components in a graph. (6)
6. Explain the various applications of Graphs.

UNIT – V

1. What is meant by Sorting?

Sorting is ordering of data in an increasing or decreasing fashion according to some linear relationship among the data items.

2. List the different sorting algorithms.

- Bubble sort
- Selection sort
- Insertion sort
- Shell sort
- Quick sort
- Radix sort
- Heap sort
- Merge sort

3. Why bubble sort is called so?

The bubble sort gets its name because as array elements are sorted they gradually "bubble" to their proper positions, like bubbles rising in a glass of soda.

4. State the logic of bubble sort algorithm.

The bubble sort repeatedly compares adjacent elements of an array. The first and second elements are compared and swapped if out of order. Then the second and third elements are compared and swapped if out of order. This sorting process continues until the last two elements of the array are compared and swapped if out of order.

5. What number is always sorted to the top of the list by each pass of the Bubble sort algorithm?

Each pass through the list places the next largest value in its proper place. In essence, each item "bubbles" up to the location where it belongs.

6. When does the Bubble Sort Algorithm stop?

The bubble sort stops when it examines the entire array and finds that no "swaps" are needed. The bubble sort keeps track of the occurring swaps by the use of a flag.

7. State the logic of selection sort algorithm.

It finds the lowest value from the collection and moves it to the left. This is repeated until the complete collection is sorted.

8. What is the output of selection sort after the 2nd iteration given the following sequence? 16 3 46 9 28 14

Ans: 3 9 46 16 28 14

7. How does insertion sort algorithm work?

In every iteration an element is compared with all the elements before it. While comparing if it is found that the element can be inserted at a suitable position, then space is created for it by shifting the other elements one position up and inserts the desired element at the suitable position. This procedure is repeated for all the elements in the list until we get the sorted elements.

7. What operation does the insertion sort use to move numbers from the unsorted section to the sorted section of the list?

The Insertion Sort uses the swap operation since it is ordering numbers within a single list.

11. How many key comparisons and assignments an insertion sort makes in its worst case?

The worst case performance in insertion sort occurs when the elements of the input array are in descending order. In that case, the first pass requires one comparison, the second pass requires two comparisons, third pass three comparisons,….kth pass requires (k-1), and finally the last pass requires (n-1) comparisons. Therefore, total numbers of comparisons are:

f(n) = 1+2+3+……+(n-k)+…..+(n-2)+(n-1) = n(n-1)/2 = O(n2)

12. Which sorting algorithm is best if the list is already sorted? Why?

Insertion sort as there is no movement of data if the list is already sorted and complexity is of the order O(N).

13. Which sorting algorithm is easily adaptable to singly linked lists? Why?

Insertion sort is easily adaptable to singly linked list. In this method there is an array link of pointers, one for each of the original array elements. Thus the array can be thought of as a linear link list pointed to by an external pointer first initialized to 0. To insert the k^{th} element the linked list is traversed until the proper position for x[k] is found, or until the end of the list is reached. At that point x[k] can be inserted into the list by merely adjusting the pointers without shifting any elements in the array which reduces insertion time.

14. Why Shell Sort is known diminishing increment sort?

The distance between comparisons decreases as the sorting algorithm runs until the last phase in which adjacent elements are compared. In each step, the sortedness of the sequence is increased, until in the last step it is completely sorted.

15. Which of the following sorting methods would be especially suitable to sort a list L consisting of a sorted list followed by a few "random" elements?

Quick sort is suitable to sort a list L consisting of a sorted list followed by a few "random" elements.

11. Which sorting algorithm follows the divide-and-conquer strategy?

Quick sort and Merge sort

12. **What is the output of quick sort after the 3rd iteration given the following sequence?** 24 56 47 35 10 90 82 31

Pass 1:- (10) 24 (56 47 35 90 82 31)
Pass 2:- 10 24 (56 47 35 90 82 31)
Pass 3:- 10 24 (47 35 31) 56 (90 82)

13. Mention the different ways to select a pivot element.

The different ways to select a pivot element are

- Pick the first element as pivot
- Pick the last element as pivot
- Pick the Middle element as pivot
- Median-of-three elements
    - Pick three elements, and find the median x of these elements
    - Use that median as the pivot.
- Randomly pick an element as pivot.

14. What is divide-and-conquer strategy?

- Divide a problem into two or more sub problems
- Solve the sub problems recursively
- Obtain solution to original problem by combining these solutions

15. Compare quick sort and merge sort.

Quicksort has a best-case linear performance when the input is sorted, or nearly sorted. It has a worst-case quadratic performance when the input is sorted in reverse, or nearly sorted in reverse.

Merge sort performance is much more constrained and predictable than the performance of quicksort. The price for that reliability is that the average case of

merge sort is slower than the average case of quicksort because the constant factor of merge sort is larger.

16. What is the key idea of radix sort?

Sort the keys digit by digit, starting with the least significant digit to the most significant digit.

17. Define Searching.

Searching for data is one of the fundamental fields of computing. Often, the difference between a fast program and a slow one is the use of a good algorithm for the data set. Naturally, the use of a hash table or binary search tree will result in more efficient searching, but more often than not an array or linked list will be used. It is necessary to understand good ways of searching data structures not designed to support efficient search.

18. What is linear search?

In Linear Search the list is searched sequentially and the position is returned if the key element to be searched is available in the list, otherwise -1 is returned. The search in Linear Search starts at the beginning of an array and move to the end, testing for a match at each item.

24. What is Binary search?

A binary search, also called a dichotomizing search, is a digital scheme for locating a specific object in a large set. Each object in the set is given a key. The number of keys is always a power of 2. If there are 32 items in a list, for example, they might be numbered 0 through 31 (binary 00000 through 11111). If there are, say, only 29 items, they can be numbered 0 through 28 (binary 00000 through 11100), with the numbers 29 through31 (binary 11101, 11110, and 11111) as dummy keys.

24. Define hash function?

Hash function takes an identifier and computes the address of that identifier in the hash table using some function.

25. Why do we need a Hash function as a data structure as compared to any other data structure? (may 10)

Hashing is a technique used for performing insertions, deletions, and finds in constant average time.

26. What are the important factors to be considered in designing the hash function? (Nov 10)

- To avoid lot of collision the table size should be prime
- For string data if keys are very long, the hash function will take long to compute.

27. What are the problems in hashing?

a. Collision

b. Overflow

28. 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.

29. 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.

30. 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.

PART B

1. Write an algorithm to implement Bubble sort with suitable example.
2. Explain any two techniques to overcome hash collision.
3. Write an algorithm to implement insertion sort with suitable example.
4. Write an algorithm to implement selection sort with suitable example.
5. Write an algorithm to implement radix sort with suitable example.
6. Write an algorithm for binary search with suitable example.
7. Discuss the common collision resolution strategies used in closed hashing system.
8. Given the input { 4371, 1323, 6173, 4199, 4344, 9679, 1989 } and a hash function of h(X)=X (mod 10) show the resulting:
    a. Separate Chaining hash table
    b. Open addressing hash table using linear probing
9. Explain Re-hashing and Extendible hashing.

10. 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. (8) (Nov 10)
11. what are the advantages and disadvantages of various collision resolution strategies? (6)