List of Practicals

Subject: DATA STRUCTURES **Paper Code: CIC-255**

Arrays

1. Insert an element at user defined position in an array of type float (unsorted).
   Description of program:
   a. Input an array of float.
   b. Ask position from the user where the new element has to be inserted.
   c. Insert the element into the array.
   d. Print the upgraded array.

2. Insert an element at user defined position in an array of type float (sorted).
   Description of program:
   a. Input an array of float.
   b. Search for the position where the new element has to be inserted.
   c. Insert the element into the array.
   d. Print the upgraded array.

3. Delete an element from user defined position in an array of type float
   Description of program:
   a. Input an array.
   b. Ask element has to be deleted.
   c. Search the position of the element.
   d. Delete the element.
   e. Print the upgraded array.

4. Perform Linear Search on an array.
   Description of program:
   a. Read an array of type integer.
   b. Input element from user for searching.
   c. Search the element by passing the array to a function and then returning the position of the element from the function else return -1 if the element is not found.
   d. Display the position where the element has been found.

5. Perform Binary Search on an array.
   Description of program:
   a. Read an array.
   b. Input element from user for searching.
   c. Search the element by passing the array to a function and then returning the position of the element from the function

d. Display the position where the element has been found.

6. Add two polynomials using array.
   Description of program:
   a. Input two polynomials from screen use array of structures.
   b. Add these polynomials and display the output stored in the array of structure form.
7. Implement sparse matrix using array.
   Description of program:
   a. Read a 2D array from the user.
   b. Store it in the sparse matrix form, use array of structures.
   c. Print the final array.

Linked List

8. Create a linked list with nodes having information about a student and perform
   a. Insert a new node at specified position.
   b. Delete of a node with the roll number of student specified.
   c. Reversal of that linked list.
9. Create doubly linked list with nodes having information about an employee and perform Insertion at front of doubly linked list deletion at end of that doubly linked list.
10. Create circular linked list having information about a college and perform Insertion at front and deletion at end.

Stack

11. Implement two stack using a single array.
12. Create a stack and perform Pop, Push, Peek and Traverse operations on the stack using Linear Linked list.
13. Convert Infix Expression to Postfix form using Stack.
14. Convert Infix Expression to Prefix form using Stack.

Queue

15. Implement insert and delete operations in a queue using an array. The array should be storing the employee numbers of the employees in the integer form. Separate functions for display, insert and delete should be designed with appropriate arguments.
16. Create a Linear Queue using Linked List and implement different operations such as Insert, Delete, and Display the queue elements.

17. Implement insertion and deletion operations on a circular queue using linked list and each node of the linked list should store information about the lab with name of the lab and number of computers in that lab. Separate functions should be designed to insert and display information in the queue.

Trees

18. Create a Binary Tree (Display using Graphics) perform Tree traversals (Preorder, Postorder, Inorder) using the concept of recursion.
19. Implement insertion, deletion and display (inorder, preorder and postorder) on binary search tree with the information in the tree about the details of a automobile (type, company, year of make).

Sorting and Hashing

20. Implement Selection Sort, Bubble Sort, Insertion sort, Merge sort, Quick sort, and Heap Sort using array as a data structure.
21. Implement the searching using hashing method.

Graphs

22. Implement the insertion in a graph and then traversal in graph using Breadth First Search.
23. Implement the insertion in a graph and then traversal in graph using Depth First Search.