Semester – III

18CSU311 DATA STRUCTURES - PRACTICAL 4H – 2C

Instruction Hours / week: L: 0 T: 0 P: 4 Marks: Internal: **40** External: **60** Total: **100**

End Semester Exam: 3 Hours

Course Objectives

- To possess intermediate level problem solving and algorithm development skills on the computer
- To able to analyze algorithms using big-Oh notation
- To understand the fundamental data structures such as lists, trees, and graphs
- To understand the fundamental algorithms such as searching, and sorting

Course Outcomes(COs)

1. Data structures and algorithms are the building blocks in computer programming.
2. This course will give students a comprehensive introduction of common data structures, and algorithm design and analysis.
3. This course also intends to teach data structures and algorithms for solving real problems that arise frequently in computer applications, and to teach principles and techniques of computational complexity.

1. Write a program to search an element from a list. Give user the option to perform Linear or Binary search. Use Template functions.
2. WAP using templates to sort a list of elements. Give user the option to perform sorting using Insertion sort, Bubble sort or Selection sort.
3. Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +).
4. Implement Doubly Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
5. Implement Circular Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list.
6. Perform Stack operations using Linked List implementation.
7. Perform Stack operations using Array implementation. Use Templates.
8. Perform Queues operations using Circular Array implementation. Use Templates.
9. Create and perform different operations on Double-ended Queues using Linked List implementation.
10. WAP to scan a polynomial using linked list and add two polynomial.
11. WAP to calculate factorial and to compute the factors of a given no. (i)using recursion, (ii) using iteration
12. (ii) WAP to display fibonaCSUi series (i)using recursion, (ii) using iteration
13. WAP to calculate GCD of 2 number (i) with recursion (ii) without recursion

14. WAP to create a Binary Search Tree and include following operations in tree: (a) Insertion (Recursive and Iterative Implementation)
(b) Deletion by copying
(c) Deletion by Merging
(d) Search a no. in BST
(e) Display its preorder, postorder and inorder traversals Recursively
(f) Display its preorder, postorder and inorder traversals Iteratively
(g) Display its level-by-level traversals
(h) Count the non-leaf nodes and leaf nodes
(i) Display height of tree
(j) Create a mirror image of tree
(k) Check whether two BSTs are equal or not
15. WAP to convert the Sparse Matrix into non-zero form and vice-versa.
16. WAP to reverse the order of the elements in the stack using additional stack.
17. WAP to reverse the order of the elements in the stack using additional Queue.
18. WAP to implement Diagonal Matrix using one-dimensional array.
19. WAP to implement Lower Triangular Matrix using one-dimensional array.
20. WAP to implement Upper Triangular Matrix using one-dimensional array.
21. WAP to implement Symmetric Matrix using one-dimensional array.
22. WAP to create a Threaded Binary Tree as per inorder traversal, and implement operations like finding the suCSUessor / predecessor of an element, insert an element, inorder traversal.
23. WAP to implement various operations on AVL Tree.