II/IV B. TECH. FIRST SEMESTER
DATA STRUCTURES LAB (Required)

Course Code : CS 3L1 **Credits: 2**
Lab Hours: 3 periods/ week **Internal assessment: 25 Marks**
Tutorial:- **Semester end examination: 50 Marks**

Prerequisites: Data Structures

Course Objectives:

1. To implement recursive functions.
2. To arrange data using different sorting techniques.
3. To implement stack, queue, linked list, tree and graph data structures.

Course Outcomes:
At the end of this course student will:

CO1) Implement different sorting and searching algorithms

CO2) Implement the stack, Queue and their applications

CO3) Implement various types of linked lists and their applications

CO4) Perform basic operations on trees and graphs and determine minimum spanning tree

Syllabus:

Implement the following exercises using 'C' Programming language.

Exercise 1

1. Write recursive program which computes the n^{th} Fibonacci number, for appropriate values of n.
2. Write recursive program for calculation of Factorial of an integer.
3. Write a program to search an element using linear and binary search with and without recursion.

Exercise 2

90, 77, 60, 99, 55, 88, 66, 32, 41, 19

1. Arrange above data set using Bubble sort
2. Arrange above data set using Selection sort

Exercise 3

26, 5, 77, 1 , 61, 11, 59, 15, 48, 19

1. Arrange above data set using insertion sort
2. Arrange above data set using Quick sort
3. Arrange above data set using Merge sort

Exercise 4

1. Implementation of stack operations using static and dynamic arrays.
2. Implementation of queue operations using static and dynamic arrays.

Exercise 5

1. Railroad cars numbered are as 0,1,2,---,n-1. Each car is brought into the stack and removed at any time. For instance, if n=3, we could move 0, move 1, move 2 and then take the cars out, producing 2,1,0. Implement application for the given problem.

2. Consider a payment counter at which the customer pays for the items purchased. Every time a customer finished paying for their items, he/she leaves the queue from the front. Every time another customer enters the line to wait, they join the end of the line. Implement the application for this problem.

Exercise 6

Implementation of single linked list.

Exercise 7

Implementation of doubly linked list

Exercise 8

3. Representation of Sparse matrix.
4. Implementation of circular linked list

Exercise 9

Implement Exercise 5 (a) using linked lists.

Exercise 10

Implement Exercise 5(b) using linked lists.

Exercise 11

A polynomial has the main fields as coefficient, exponent in linked list it will have one more field called link to point to next term in the polynomial. If there are n terms in the polynomial then n such nodes has to be created.

Exercise 12

Implementation of binary tree: creation, insertion, deletion, traversing

Exercise 13

Implementation of Binary Search Tree operations

Exercise 14

Implementation of Graph traversals

Exercise 15

Implementation of minimum spanning tree