ASSIGNMENTS
(Practical)

Program : BCA Semester : 3rd
Course : Data Structure

Instructions :

- All the Questions Carry 5 Marks Each (Total Marks 200)
- Attempt Any 8 Questions from each module.
- All the Codes shall be uploaded in the github repository of the Students whose link has been submitted by the Students.
- Students Shall create a Repository Named “DataStructure_Practicals”. The Naming convention should be followed as ModuleNumber_ProgramNumber (ex. M1_1.cpp, M1_2.cpp ... M3_1 and so on.)

Module I: Introduction to Data Structures and Arrays

1. Write a program to declare and print elements of a 1D array of 5 integers.
2. Write a program to input 10 numbers in an array and display them.
3. Write a program to find the sum of all elements in a 1D array.
4. Write a program to find the maximum and minimum element in an array.
5. Write a program to reverse the elements of a 1D array.
6. Write a program to input a 2D array (3x3) and display it in matrix form.
7. Write a program to perform element-wise addition of two 1D arrays.
8. Write a program to count even and odd numbers in an array.
9. Write a program to copy elements of one array into another.
10. Write a program to insert an element at a specific position in a 1D array.

Module II: Stacks and Queues

1. Write a program to implement a stack using arrays (push and pop).
2. Write a program to display top element of a stack without removing it.
3. Write a program to check if a stack is empty or full.
4. Write a program to implement a queue using arrays (enqueue and dequeue).
5. Write a program to display front and rear elements of a queue.
6. Write a program to implement circular queue operations.
7. Write a program to convert infix expression to postfix expression (basic).
8. Write a program to evaluate a postfix expression using stack.
9. Write a program to check for balanced parentheses using stack.
10. Write a menu-driven program to perform stack operations using array.

Module III: Linked Lists

1. Write a program to create a singly linked list and display it.
2. Write a program to insert a node at the beginning of a singly linked list.

3. Write a program to insert a node at the end of a singly linked list.
4. Write a program to delete a node from the beginning of a singly linked list.
5. Write a program to delete a node from the end of a singly linked list.
6. Write a program to search an element in a linked list.
7. Write a program to reverse a singly linked list.
8. Write a program to create and display a doubly linked list.
9. Write a program to insert and delete nodes in a circular linked list.
10. Write a program to count the number of nodes in a linked list.

Module IV: Trees

1. Write a program to create a simple binary tree and display root, left and right nodes.
2. Write a program to perform in-order traversal of a binary tree.
3. Write a program to perform pre-order traversal of a binary tree.
4. Write a program to perform post-order traversal of a binary tree.
5. Write a program to insert a node in a Binary Search Tree (BST).
6. Write a program to search for a node in a BST.
7. Write a program to find the minimum and maximum element in a BST.
8. Write a program to delete a node from a BST.
9. Write a program to find the height of a binary tree.
10. Write a program to implement rotations in an AVL Tree (left rotation only).

Module V: Graphs and Sorting

1. Write a program to represent a graph using an adjacency matrix.
2. Write a program to represent a graph using an adjacency list.
3. Write a program to perform BFS traversal on a graph.
4. Write a program to perform DFS traversal on a graph.
5. Write a program to implement hashing with linear probing.
6. Write a program to implement a basic hash function (modulo method).
7. Write a program to sort an array using bubble sort.
8. Write a program to sort an array using selection sort.
9. Write a program to sort an array using insertion sort.
10. Write a program to implement merge sort on an array.