**International Centre for Applied Sciences**

**LAB MANUAL**

**DATA STRUCTURES with C++**

**INSTRUCTIONS TO STUDENTS**

1. Students should be regular and come prepared for the lab practice.
2. Students who turn up late to the labs will in no case be permitted to the program schedule for the day.
3. In case a student misses a class, it is his/her responsibility to complete that missed experiment(s).
4. Students should bring the observation book. Prescribed textbook and class notes can be kept ready for reference if required.
5. Students are required to prepare thoroughly to perform the experiment before coming to laboratory.
6. They should implement the given experiment individually.
7. While conducting the experiments students should see that their programs would meet the following criteria:
   1. Programs should be interactive with appropriate prompt messages, error messages if any, and descriptive messages for outputs.
   2. Programs should perform input validation (Data type, range error, etc.) and give appropriate error messages and suggest corrective actions.
   3. Comments should be used to give the statement of the problem and every function should indicate the purpose of the function, inputs and outputs.
   4. Statements within the program should be properly indented.
   5. Use meaningful names for variables and functions.
   6. Make use of Constants and type definitions wherever needed.
8. After completion of the program, certification of the concerned staff in-charge in the observation book is necessary.
9. When the experiment for the week is completed, should shut down the system.
10. Students should be present in the labs for the total scheduled duration.
11. Roaming in the lab/discussions inside the lab is strictly prohibited.
12. Questions for lab tests and exams need not necessarily be limited to the questions in the manual, but could involve some variations and/or combinations of the questions.

**CONTENTS**

SL NO.                            TITLE  OF EXPERIMENT                            PAGE NO.

1.                         Review of C++         4

2.                                    Review of C++ and Templates         4

3.  Stacks      5

4.    Recursion 5

5.                               Queues                          6

6.                               Lists                      6

7.                               Lists                          6

8.                               Trees                    7

9.                              Trees                7

10. Sorting 7

11. Graph 7

**Week 1 -Review of C++**

**Write a program in C++ to:**

1. Find sum of given n numbers using a 1D array. Use a function to find the sum of elements in the array and main program to display the sum.
2. Sort given list of n integers into ascending order using selection sort. Use function to sort.
3. Implement Linear Search on list of integers.
4. Implement Binary search on list of integers.
5. Find sum of 2 matrices using a function named add with suitable parameters.
6. Find biggest in a list of numbers using a function.
7. To add two numbers using pointers.
8. To
   1. Demonstrate passing pointers to a function.
   2. Demonstrate Returning pointer from a function.
   3. Using pointer to pointer.

**Note: Debug above programs using debugging features in TCPLUS.(F4,F7,F8,CTRL+  
F4,CTRL+F7,CTRL+F2 etc)**

**Week2 -Review of C++** **and Templates**

Write a program in C++ using class for the following:

1. To find roots of quadratic equation:

Data members: a,b,c

Member functions:

a) getdata to read coefficients

b) ComputeRoots to find roots.

c) putdata to print the roots.

Write a main function to test the class.

1. To perform complex number arithmetic on 2 complex numbers.
2. Write a function template **Swap** to swap the values of 2 variables. Instantiate this function template for integers, floats and characters.
3. *Write function templates,* ***Sort,*** *which will sort a given list of values and* ***PrintArray,*** *which prints the list. Write a main program, which will input, sort and print an int array and float array.*
4. Write a class template Arithmetic, with 2 data members and member functions to read, display, find smaller and larger of the members. Instantiate this template for integer and floating point data.
5. Write a program to sort an array of student structures according to the roll number.

**Week 3 -Stacks**

**Write C++ program to:**

1. Implement Stack using an array. The elements of the stack are characters. Include functions push, pop and display. Also include functions for checking error conditions such as underflow and overflow.
2. Check whether the given string is palindrome using stack.
3. Convert a given decimal number to binary using stack.
4. Check if the given parenthesized expression has properly matching open and closing parenthesis using stack.
5. *Evaluate the given postfix expression using stack.*
6. *Convert a fully parenthesized infix expression to postfix.*

**Week 4 - Recursion**

**Write a recursive program in C++ to:**

1) Find factorial of a given number.

2) Find GCD of 2 numbers.

*3) Display Fibonacci series upto Nth term.*

*4) Simulate the working of Tower of Hanoi for n disks. Print the number of moves.*

5) Convert given decimal number to binary.

**Week 5 -Queues**

**Write C++ program to:**

*1) Implement a queue of characters. Include functions insertq, deleteq and displayq.*

*2) Implement a circular queue of integers. Include functions insert, delete and display.*

3) Implement an ascending priority queue.

**Week 6 - 7 -Linked List**

**Write C++ program to:**

1. Implement stack using singly linked list.
2. Implement queues using singly linked list.
3. *Implement the following operations on Singly Linked List:*
4. Insert at the front
5. Insert at the last
6. Search for a given item
7. Delete an item
8. Display the list.
9. Reversing the list
10. *Implement the following operations on Doubly Linked List:*
11. Insert at the front
12. Insert at the last
13. Search for a given item
14. Delete an item
15. Display the list.
16. Reversing the list
17. Implement Union and Intersection using doubly linked list.

**Week 8 - 9 –Trees**

**Write C++ program to:**

1. *Create a simple binary tree using recursion*
2. Create a tree, traverse and search for an item.
3. *Display inorder, preoder and postorder traversal on the binary tree created using recursion.*
4. Check if two trees are equal
5. Write a program to insert, delete in a BST
6. *Implement level order traversal on BST*
7. Create a tree for postfix expression and evaluate it

**Week 10 – Sorting**

**Write C++ program to:**

1. Write a program to sort numbers using merge sort
2. Write a program to sort numbers using Quick sort
3. Write a program to sort numbers using bubble sort

**Week 11 –Graphs**

**Write C++ program to:**

1. Write a program to implement Depth First Search on a graph.
2. Write a program to implement Breadth First Search on a graph.

**References:**

1. Fundamentals of Data Structures Using C++, Sahani, Horowitz and Dinesh Mehta, Galgotia Publications.
2. Data Structures and Problem Solving Using C++, Mark Allen Weiss, 2nd Edition, Pearson Education International Inc.
3. Fundamentals of Data Structures, Horowitz and Sahani