Data Structures Lab Manual

Lab 3: Iterators & Circular Linked List

Prepared for: Students of Data Structures

Department of Computer Science  
Fast School of Computing

# Objectives

1. To understand and implement the concept of iterators in C++
2. To explore how iterators help in sequentially accessing elements of data structures.
3. To design and implement a Circular Linked List and perform operations on it.
4. To enhance problem-solving and logical reasoning skills through challenging exercises.

# Lab Outcomes

After completing this lab, students will be able to:

1. Use iterators to traverse through standard containers like list
2. Differentiate between standard indexing and iterator-based traversal.
3. Implement and manipulate a Circular Linked List.
4. Perform insertion, deletion, traversal, and searching on Circular Linked Lists.
5. Apply critical thinking to design custom solutions for real-world inspired problems.

# Lab Task

## Part A: Iterators

1. Iterator Basics:  
    - Write a program to store integers in a vector<int>.  
    - Use an iterator to print all elements.  
    - Repeat using a reverse iterator.
2. Iterator with STL Containers:  
    - Store student names in a list<string>.  
    - Use iterators to:  
    a) Print all names.  
    b) Erase a name using an iterator.  
    c) Insert a new name at a specific position.
3. Set Iterator:  
    - Create a set<int> of random numbers.  
    - Use an iterator to print the sorted unique elements.  
    - Compare iteration vs indexing.

## Part B: Circular Linked List

1. Creating Circular Linked List:  
    - Define a Node structure with data and next pointer.  
    - Write a function to insert a node at the end of a circular linked list.  
    - Traverse the list and print elements using a loop.
2. Insertion at Beginning and End:  
    - Extend the previous program to allow:  
    a) Insert at the beginning.  
    b) Insert at the end.
3. Deletion in Circular Linked List:  
    - Implement deletion of:  
    a) First node.  
    b) Last node.  
    c) A node with a specific key.
4. Menu-Driven Program:  
    - Write a menu-driven program to perform:  
    - Insertion at beginning/end.  
    - Deletion.  
    - Traversal.  
    - Search operation.

## Part C: Josephus Problem

- The Josephus Problem is a theoretical problem related to a certain elimination game.  
- Statement: N people are standing in a circle. Starting from the first person, every k-th person is eliminated until only one remains.

1. Task:
2. Implement this using a Circular Linked List.
3. Write a program that takes N (number of people) and k (step count) as input and prints the position of the last remaining person.  
     
   **Hint: This problem models real-world CPU process scheduling and is widely used in algorithmic competitions.**

# Submission Guidelines

- Submit your .cpp file with proper comments.  
- Make sure your program compiles and runs successfully.