Data Structures Lab Manual

Lab 3: Doubly link list

Prepared for: Students of Data Structures

Department of Computer Science  
Fast School of Computing

# Objectives

- To understand the structure and functionality of Doubly Linked Lists (DLL).  
- To implement insertion, deletion, traversal, and search operations in DLL.  
- To analyze advantages of DLL over singly linked list.  
- To apply DLL in solving real-world style problems.

# Lab Outcomes

After completing this lab, students will be able to:  
1. Define the structure of a DLL node and explain its working.  
2. Implement different insertion and deletion operations.  
3. Traverse a DLL in forward and backward directions.  
4. Apply DLL to real-world use cases (e.g., playlist, undo/redo operations).  
5. Compare DLL with singly linked list in terms of efficiency and flexibility.

# Lab Task

### Task 1: Create a Node Structure

- Define a Node class for DLL with data, prev, and next.  
- Write a small main() to create 3 nodes manually and link them forward and backward.  
- Display the elements in both forward and backward directions.

### Task 2: Insertion Operations

Write functions to insert nodes in a DLL:  
1. Insert at the beginning  
2. Insert at the end  
3. Insert at a given position (e.g., 2nd or 3rd position)  
  
👉 Test Case: Start with an empty DLL and insert 10, 20, 30, 40. Display forward and backward traversal after each insertion.

### Task 3: Deletion Operations

Write functions to delete nodes in DLL:  
1. Delete from the beginning  
2. Delete from the end  
3. Delete a specific node by value  
  
👉 Test Case: Create DLL: 10 <-> 20 <-> 30 <-> 40.  
- Delete first node, print list.  
- Delete last node, print list.  
- Delete node with value 20, print list.

### Task 4: Traversal & Search

- Write a function to traverse forward and backward.  
- Write a function to search for a given value in DLL and return its position.  
  
Test Case: On DLL 10 <-> 20 <-> 30 <-> 40, search for 30 (found at position 3) and 50 (not found).

### Task 5: Application – Playlist Manager

Implement a simple playlist system using DLL where each node stores a song name.  
Features:  
1. Add a new song at the end.  
2. Delete a song by name.  
3. Show playlist forward.  
4. Show playlist backward (like 'previous song' feature).  
  
Test Case: Add songs "Song1", "Song2", "Song3". Delete "Song2". Show playlist forward and backward.

### Task 6 (Optional ): Compare SLL vs DLL

- Write a short note or small code snippet showing why DLL is more powerful than SLL in backward traversal and deletion at arbitrary nodes.

# Submission Guidelines

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