



Aror University of Art, Architecture, Design & Heritage Sukkur

Department of AI-Multimedia and Gaming

Lab 01: Fundamentals of Doubly Linked List data structure

Date: Sep 12, 2024

Subject: Data Structure (CSC221), Fall 2024

Instructor: Abdul Ghafoor

Lab objectives: To practice and understand the basic operations in a doubly linked list, including insertion, deletion, Searching, and traversal of nodes.

Lab Part 01

Instructions:

You will implement a doubly linked list in Java (or any other programming language of your choice).

Perform the following tasks, ensuring each method works correctly and efficiently.

Node Structure Implementation

- Define a Node class that will represent each element in the linked list.
- Each Node should contain an integer data field and a reference to the next and prev nodes in the list.

Doubly Linked List Class

- Create a doubly LinkedList class that will contain the following operations:

a) Insertion Operations:

- **Insert at Start:** Write a method `addAtStart(int data)` that inserts a new node at the start of the linked list.
- **Insert at End:** Write a method `addAtEnd(int data)` that inserts a new node at the end of the linked list.
- **Insert at Position:** Write a method `addAtPosition(int data, int position)` that inserts a new node at a specific position in the list. If the position is invalid (greater than the length of the list), print an appropriate message.

b) Deletion Operations:

Delete from Start: Write a method deleteFromStart() that deletes the first node of the linked list.

Delete from End: Write a method deleteFromEnd() that deletes the last node of the linked list.

Delete by Value: Write a method deleteByValue(int data) that deletes the first occurrence of a node with the given value from the linked list. If the value is not found, print an appropriate message.

c) Traversal Operations:

- **Display List:** Write a methods displayFromStart() and displayFromEnd() that traverses the linked list and prints each node's data.

d) Search Operation:

- **Search for a Value:** Write a method search(int value) that searches for a node with the given value in the linked list. If found, print the position (0-based index), otherwise print "Value not found."

e) **Reverse the List:** Write a method reverse() that reverses the linked list.

f) **Count Nodes:** Write a method countNodes() that returns the total number of nodes in the linked list.

Testing and Validation:

- Implement a main method where you:
- Insert nodes at both the start and end of the list.
- Insert nodes at specific positions in the list.
- Delete nodes from both the start and end of the list.
- Search for values in the list.
- Display the list after each operation to verify correctness.

Lab Part 02: LeetCode

Task 01:

<https://leetcode.com/problems/middle-of-the-linked-list/description/?envType=problem-list-v2&envId=linked-list>

Task 02:

<https://leetcode.com/problems/convert-binary-number-in-a-linked-list-to-integer/?envType=problem-list-v2&envId=linked-list>

Task 03:

<https://leetcode.com/problems/add-two-numbers/description/?envType=problem-list-v2&envId=linked-list>

Due Date: 18 Sep, 2024