Lab Report-6

Course Title :Data Structure

Course Code: CSE207

Course Instructor :Dr. Anup Kumar Paul

Semester:Spring 2024

Section:03

Experiment Name:A program to create a Linked List and perform operations such as insert, and delete.

Submitted by-

Name:Nuran Farhana Prova

ID: 2023-1-60-075

Source Code:

Main Class:

```
package LinkedlistOperations;
public class Main {
	public static void main(String[] args) {
		LinkedListOperations list = new LinkedListOperations();
		list.insertAtFirst();
		list.insertAtFirst();
		list.insertAtFirst();
		System.out.print("Before inserting at first node\n");
		list.printList();
		System.out.print("\n");
		list.insertAtFirst();
		System.out.println("After inserting at first node");
		list.printList();
		System.out.print("\n");
		list.insertAtMiddle();
		System.out.println("After inserting at middle node");
		list.printList();
		System.out.print("\n");
		list.insertAtLast();
		System.out.println("\nAfter inserting at last node");
		list.printList();
		list.deleteFirst();
		System.out.println("\nAfter deleting first node");
		list.printList();
		System.out.print("\n");
		list.deleteMiddle();
		System.out.println("After deleting middle node");
		list.printList();
		list.deleteLast();
		System.out.println("\nAfter deleting last node");
		list.printList();
		System.out.println("\nFinally displaying the list:");
		list.printList();
	}
}
```

LinkedList Class:

```
package LinkedlistOperations;
import java.util.Scanner;
public class LinkedListOperations {
        Node start;
        public LinkedListOperations() {
                start = null;
        }
        public Node getNode() {
                Node newNode = new Node();
                Scanner input = new Scanner(System.in);
                System.out.println("Enter data");
                newNode.data = input.nextInt();
                newNode.next = null;
                return newNode;
        }
        public void insertAtFirst() {
                Node newNode = getNode();
                if (start == null) {
                        start = newNode;
                } else {
                        newNode.next = start;
                        start = newNode;
                }
        }
        public int nodeCounter() {
                Node temp = start;
                int count = 1;
                while (temp.next != null) {
                        temp = temp.next;
                        count++;
                }
                return count;
        }
        public void insertAtMiddle() {
                Node newNode = getNode();
                if (start == null) {
                        start = newNode;
                } else {
                        Scanner input = new Scanner(System.in);
                        System.out.println("Enter the position");
        int position = input.nextInt();
        if (position > 1 && position <= nodeCounter()) {
            Node temp = start;
            int ctr = 1;
            while (ctr < position - 1) {
                temp = temp.next;
                ctr++;
            }
            newNode.next = temp.next;
            temp.next = newNode;
        } else {
            System.out.println("Invalid position");
        }
    }
}
public void insertAtLast() {
    Node newNode = getNode();
    if (start == null) {
        start = newNode;
    } else {
        Node temp = start;
        while (temp.next != null) {
            temp = temp.next;
        }
        temp.next = newNode;
    }
}
public void deleteFirst() {
    if (start == null) {
        System.out.println("Empty list..nothing to delete");
    } else {
        start = start.next;
    }
}
public void deleteMiddle() {
    if (start == null) {
        System.out.println("Empty list");
    } else {
        Scanner input = new Scanner(System.in);
        System.out.println("Enter the position");
        int position = input.nextInt();
        if (position > 1 && position <= nodeCounter()) {
```

```
                    Node temp = start;
                    int ctr = 1;
                    while (ctr < position - 1) {
                            temp = temp.next;
                            ctr++;
                    }
                    temp.next = temp.next.next;
                } else {
                    System.out.println("Invalid position");
                }
            }
        }
        public void deleteLast() {
            if (start == null) {
                System.out.println("Empty List");
            } else {
                Node temp = start;
                while (temp.next.next != null) {
                    temp = temp.next;
                }
                temp.next = null;
            }
        }
        public void printList() {
            Node temp = start;
            while (temp != null) {
                if (temp.next != null) {
                    System.out.print(temp.data + "-->");
                } else {
                    System.out.print(temp.data);
                }
                temp = temp.next;
            }
        }
}
```

Node Class:

```
package LinkedlistOperations;
public class Node {
        int data;
        Node next;
}
```

Output :

```
Enter data
10
Enter data
20
Enter data
50
Before inserting at first node
50-->20-->10
Enter data
100
After inserting at first node
100-->50-->20-->10
Enter data
45
Enter the position
3
After inserting at middle node
100-->50-->45-->20-->10
Enter data
8

After inserting at last node
100-->50-->45-->20-->10-->8
After deleting first node
50-->45-->20-->10-->8
Enter the position
2
After deleting middle node
50-->20-->10-->8
After deleting last node
50-->20-->10
Finally displaying the list:
50-->20-->10
```