Program No.	07
Roll No.	1525
Topic.	Circular Linked List
Title of Program.	
Objective.	

Source Code:

SLL.java

```
/*      Name: Bhairav kedare
        Roll No: 1525
        Unit 4: List
        Program: CircularLinked List
*/



import java.util.Scanner;

// 1. Node Class
class Node {
    int data;
    Node next;

    public Node(int d) {
        data = d;
        next = null;
    }
} // end of Node class

// 2. List Class
class List {
    Node head;
    Node tail;
    int size;

    // Constructor
    public List() {
        head = null;
        tail = null;
        size = 0;
    }
```

```
// Return size of the list
public int getSize() {
    return size;
}

// Return node at the end of the list
public void InsertEnd(int val) {
    Node x = new Node(val);
    size++;

    if (head == null) { // empty list
        head = x;
        tail = x;
    } else {
        tail.next = x;
        tail = x;
    }
} // end of InsertEnd

// Insert at the Head
public void InsertStart(int val) {
    Node x = new Node(val);
    size++;

    if (head == null) { // empty list
        head = x;
        tail = x;
    } else {
        x.next = head;
        head = x;
    }
} // end of InsertStart

        /*count the number*/
        public void Count()
        {
            Node tmp =head;
            int count = 0;

            while(tmp!=null)
            {
                count++;
                tmp = tmp.next;
            }
            System.out.println(count);
        }


// Display the list
public void Display() {
    Node tmp = head; // Initialize tmp to the first node
```

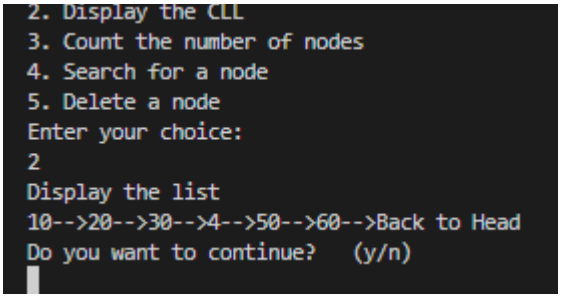
```
    while (tmp != null) {
        System.out.print(tmp.data + " -> "); // Print data at tmp
        tmp = tmp.next; // Shift tmp to the next node
    }
    System.out.println("null");
} // end of Display

// Search Node
public void Search(int val) {
    Node tmp = head;
    boolean found = false;

    while (tmp != null) {
        if (tmp.data == val) { // Search value
            found = true;
            break;
        }
        tmp = tmp.next;
    } // end of while

    if (found) {
        System.out.println(val + " is Found");
    } else {
        System.out.println(val + " is not Found");
    }
} // end of Search

// Deletion of node
public void Del(int val) {
    Node tmp = head;
    Node prev = null;
    boolean found = false;

    while (tmp != null) {
        if (tmp.data == val) {
            found = true;
            break;
        }
        prev = tmp;
        tmp = tmp.next;
    } // end of while

    // Unsuccessful Search
    if (!found) {
        System.out.println("Unsuccessful Search!");
        return;
    }

    // Successful search
    if (tmp == head && tmp == tail) { // Single node deletion
        head = null;
        tail = null;
```

```
        } else if (tmp == head) { // Head node deletion
            head = tmp.next;
        } else if (tmp == tail) { // Tail node deletion
            tail = prev;
            tail.next = null;
        } else { // Any node deletion in middle
            prev.next = tmp.next;
        }
        size--; // Decrement size
    } // end of Del
} // end of List class

// 3. Interface class SLL
public class SLL {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        List s = new List();
        char ch;

        do {
            System.out.println("*** Singly Linked List ***");
            System.out.println("1. Insert at the end of the list");
            System.out.println("2. Size of the list");
            System.out.println("3. Display the list");
            System.out.println("4. Search in the list");
            System.out.println("5. Insert at the Start of the list");
            System.out.println("6. Deletion of Node");
            System.out.print("Enter your choice: ");
            int choice = scan.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter a value at end: ");
                    s.InsertEnd(scan.nextInt());
                    break;
                case 2:
                    System.out.println("Size: " + s.getSize() + "\n");

                    break;
                case 3:
                    System.out.print("SLL contains: ");
                    s.Display();
                    break;
                case 4:
                    System.out.print("Value to Search: ");
                    s.Search(scan.nextInt());
                    break;
                case 5:
                    System.out.print("Enter a value to insert at start: ");
                    s.InsertStart(scan.nextInt());
                    break;
                case 6:
```

```
                System.out.print("Enter a value for deletion: ");
                s.Del(scan.nextInt());
                break;
            default:
                System.out.println("Incorrect Choice");
        } // end of switch

        System.out.print("Do you want to continue? (Type: Y or N) ");
        ch = scan.next().charAt(0);
    } while (ch == 'y' || ch == 'Y'); // end of do while
  } // end of main
} // end of class SLL
```

Output Screenshot

Insert at end of list



Search Node Found

```
Display the list:
10 --> 20 --> 30 --> 40 --> 50 --> Back to Head
Do you want to continue? (y/n): 1
1. Insert in Circular Linked List
2. Display the CLL
3. Count the number of nodes
4. Search for a node
5. Delete a node
Enter your choice: 4
Enter value to search for: 10
Data found: 10
Do you want to continue? (y/n):
```

Not Found

```
10 --> 20 --> 30 --> 40 --> 50 --> Back to Head
Do you want to continue? (y/n): 1
1. Insert in Circular Linked List
2. Display the CLL
3. Count the number of nodes
4. Search for a node
5. Delete a node
Enter your choice: 4
Enter value to search for: 99
Data not found: 99
Do you want to continue? (y/n):
```