```java
// Q1 Circular Linked List
import java.util.*;
class Q1Circular {
    static class Node{int data;Node next;Node(int d){data=d;}}
    static Node head=null;
    static void insertFirst(int x){
        Node n=new Node(x);
        if(head==null){head=n;n.next=n;return;}
        Node t=head;
        while(t.next!=head)t=t.next;
        n.next=head;
        head=n;
        t.next=head;
    }
    static void insertLast(int x){
        Node n=new Node(x);
        if(head==null){head=n;n.next=n;return;}
        Node t=head;
        while(t.next!=head)t=t.next;
        t.next=n;
        n.next=head;
    }
    static void insertAfter(int key,int x){
        if(head==null)return;
        Node t=head;
        do{
            if(t.data==key){
                Node n=new Node(x);
                n.next=t.next;
                t.next=n;
                return;
            }
            t=t.next;
        }while(t!=head);
    }
    static void insertBefore(int key,int x){
        if(head==null)return;
        if(head.data==key){insertFirst(x);return;}
        Node prev=head,cur=head.next;
        while(cur!=head){
            if(cur.data==key){
                Node n=new Node(x);
                n.next=cur;
                prev.next=n;
                return;
            }
            prev=cur;
            cur=cur.next;
        }
    }
    static void deleteVal(int key){
        if(head==null)return;
        if(head.data==key){
            if(head.next==head){head=null;return;}
            Node t=head;
```

```
            while(t.next!=head)t=t.next;
            head=head.next;
            t.next=head;
            return;
        }
        Node prev=head,cur=head.next;
        while(cur!=head){
            if(cur.data==key){
                prev.next=cur.next;
                return;
            }
            prev=cur;
            cur=cur.next;
        }
    }
    static boolean search(int key){
        if(head==null)return false;
        Node t=head;
        do{
            if(t.data==key)return true;
            t=t.next;
        }while(t!=head);
        return false;
    }
    static void display(){
        if(head==null)return;
        Node t=head;
        do{System.out.print(t.data+" ");t=t.next;}while(t!=head);
        System.out.println();
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        while(sc.hasNextInt()){
            int ch=sc.nextInt();
            if(ch==0)break;
            if(ch==1){int x=sc.nextInt();insertFirst(x);}
            else if(ch==2){int x=sc.nextInt();insertLast(x);}
            else if(ch==3){int
k=sc.nextInt(),x=sc.nextInt();insertAfter(k,x);}
            else if(ch==4){int
k=sc.nextInt(),x=sc.nextInt();insertBefore(k,x);}
            else if(ch==5){int k=sc.nextInt();deleteVal(k);}
            else if(ch==6){int
k=sc.nextInt();System.out.println(search(k)?"FOUND":"NOTFOUND");}
            else if(ch==7){display();}
        }
        sc.close();
    }
}

// Doubly Linked List
import java.util.*;
class Q1Doubly {
    static class Node{int data;Node prev,next;Node(int d){data=d;}}
    static Node head=null;
```

```
    static void insertFirst(int x){
        Node n=new Node(x);
        if(head==null){head=n;return;}
        n.next=head;
        head.prev=n;
        head=n;
    }
    static void insertLast(int x){
        Node n=new Node(x);
        if(head==null){head=n;return;}
        Node t=head;
        while(t.next!=null)t=t.next;
        t.next=n;
        n.prev=t;
    }
    static void insertAfter(int key,int x){
        Node t=head;
        while(t!=null){
            if(t.data==key){
                Node n=new Node(x);
                n.next=t.next;
                n.prev=t;
                if(t.next!=null)t.next.prev=n;
                t.next=n;
                return;
            }
            t=t.next;
        }
    }
    static void insertBefore(int key,int x){
        if(head==null)return;
        if(head.data==key){insertFirst(x);return;}
        Node t=head.next;
        while(t!=null){
            if(t.data==key){
                Node n=new Node(x);
                n.next=t;
                n.prev=t.prev;
                t.prev.next=n;
                t.prev=n;
                return;
            }
            t=t.next;
        }
    }
    static void deleteVal(int key){
        if(head==null)return;
        if(head.data==key){
            head=head.next;
            if(head!=null)head.prev=null;
            return;
        }
        Node t=head.next;
        while(t!=null){
            if(t.data==key){
```

```
                if(t.prev!=null)t.prev.next=t.next;
                if(t.next!=null)t.next.prev=t.prev;
                return;
            }
            t=t.next;
        }
    }
    static boolean search(int key){
        Node t=head;
        while(t!=null){
            if(t.data==key)return true;
            t=t.next;
        }
        return false;
    }
    static void display(){
        Node t=head;
        while(t!=null){System.out.print(t.data+" ");t=t.next;}
        System.out.println();
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        while(sc.hasNextInt()){
            int ch=sc.nextInt();
            if(ch==0)break;
            if(ch==1){int x=sc.nextInt();insertFirst(x);}
            else if(ch==2){int x=sc.nextInt();insertLast(x);}
            else if(ch==3){int
k=sc.nextInt(),x=sc.nextInt();insertAfter(k,x);}
            else if(ch==4){int
k=sc.nextInt(),x=sc.nextInt();insertBefore(k,x);}
            else if(ch==5){int k=sc.nextInt();deleteVal(k);}
            else if(ch==6){int
k=sc.nextInt();System.out.println(search(k)?"FOUND":"NOTFOUND");}
            else if(ch==7){display();}
        }
        sc.close();
    }
}


// Q2. Display Circular List with head repeated at end
import java.util.*;
class Q2DisplayCircular {
    static class Node{int data;Node next;Node(int d){data=d;}}
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        Node head=null,tail=null;
        for(int i=0;i<n;i++){
            int x=sc.nextInt();
            Node node=new Node(x);
            if(head==null){head=tail=node;}
            else{tail.next=node;tail=node;}
        }
```

```
        if(head!=null)tail.next=head;
        if(head==null){sc.close();return;}
        Node t=head;
        do{System.out.print(t.data+" ");t=t->next;}while(t!=head);
        System.out.print(head.data);
        sc.close();
    }
}


// Q3. Size of Doubly Linked List
import java.util.*;
class Q3SizeDLL {
    static class Node{int data;Node prev,next;Node(int d){data=d;}}
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        Node head=null,tail=null;
        for(int i=0;i<n;i++){
            int x=sc.nextInt();
            Node node=new Node(x);
            if(head==null){head=tail=node;}
            else{tail.next=node;node.prev=tail;tail=node;}
        }
        int c=0;Node t=head;
        while(t!=null){c++;t=t.next;}
        System.out.println(c);
        sc.close();
    }
}


// Q3(ii) Size of Circular Linked List
import java.util.*;
class Q3SizeCLL {
    static class Node{int data;Node next;Node(int d){data=d;}}
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        Node head=null,tail=null;
        for(int i=0;i<n;i++){
            int x=sc.nextInt();
            Node node=new Node(x);
            if(head==null){head=tail=node;}
            else{tail.next=node;tail=node;}
        }
        if(head!=null)tail.next=head;
        if(head==null){System.out.println(0);sc.close();return;}
        int c=0;Node t=head;
        do{c++;t=t.next;}while(t!=head);
        System.out.println(c);
        sc.close();
    }
}
```

```java
// Q4. Check if Doubly Linked List of chars is palindrome
import java.util.*;
class Q4PalDLL {
    static class Node{char data;Node prev,next;Node(char d){data=d;}}
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        String s=sc.next();
        Node head=null,tail=null;
        for(int i=0;i<s.length();i++){
            Node node=new Node(s.charAt(i));
            if(head==null){head=tail=node;}
            else{tail.next=node;node.prev=tail;tail=node;}
        }
        Node l=head,r=tail;
        boolean ok=true;
        while(l!=null && r!=null && l!=r && r.next!=l){
            if(l.data!=r.data){ok=false;break;}
            l=l.next;r=r.prev;
        }
        System.out.println(ok?"YES":"NO");
        sc.close();
    }
}

// Q5. Check if Linked List is Circular
import java.util.*;
class Q5IsCircular {
    static class Node{int data;Node next;Node(int d){data=d;}}
    static boolean isCircular(Node head){
        if(head==null)return false;
        Node t=head.next;
        while(t!=null && t!=head)t=t.next;
        return t==head;
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        Node head=null,tail=null;
        for(int i=0;i<n;i++){
            int x=sc.nextInt();
            Node node=new Node(x);
            if(head==null){head=tail=node;}
            else{tail.next=node;tail=node;}
        }
        int makeCirc=sc.nextInt();
        if(makeCirc==1 && tail!=null)tail.next=head;
        System.out.println(isCircular(head)?"CIRCULAR":"NOTCIRCULAR");
        sc.close();
    }
}
```