CIRCULAR LINKED LIST

public static class CLL {  
  
  
 int size;  
  
 CLL() {  
 this.size = 0;  
 }  
  
 class Node {  
 int val;  
 Node next;  
  
  
 Node(int val) {  
 this.val = val;  
 size++;  
 }  
 }  
  
 Node head;  
 Node tail;  
  
 public void addFirst ( int val){  
 Node newn = new Node(val);  
 if (head == null) {  
 head=newn;  
 tail=newn;  
 }  
 else{  
 newn.next = head;  
 tail.next= newn;  
 head = newn;  
 }  
  
 }  
  
 public void addLast ( int val){  
 Node newn = new Node(val);  
 if (head == null) {  
 addFirst(val);  
 }  
 Node curr = head;  
 while (curr!= tail) {  
 curr = curr.next;  
 }  
 curr.next = newn;  
 newn.next=head;  
 tail=newn;  
  
 }  
 public void insert(int index,int val){  
 if(head==null){  
 addFirst(val);  
 }  
 Node newn=new Node(val);  
 Node curr=head;  
 for(int i=1;i<index-1;i++){  
 curr=curr.next;  
  
 }  
 newn.next=curr.next;  
 curr.next=newn;  
  
 }  
 public void delFirst(){  
 if(head==null){  
 return;  
 }  
 size--;  
 if(head==tail){  
 head=null;  
 }  
 head=head.next;  
 tail.next=head;  
 }  
 public void delLast(){  
 if(head==null){  
 return;  
 }  
 size--;  
 if(head==tail){  
 head=null;  
 }  
 Node curr=head;  
 while(curr.next!=tail){  
 curr=curr.next;  
 }  
 curr.next=head;  
 tail=curr;  
 }  
 public void delete(int index){  
 if(head==null){  
 return;  
 }  
 size--;  
 if(head==tail){  
 head=null;  
 }  
 Node curr=head;  
 for(int i=1;i<index-1;i++){  
 curr=curr.next;  
  
 }  
 curr.next=curr.next.next;  
  
 }  
 public void rev() {  
 if(head==null){  
 return;  
 }  
 if(head==tail){  
 System.*out*.println("same");  
 }  
 Node prev = head;  
 Node curr = head.next;  
 Node nextn = null;  
 while (curr != head) {  
 nextn = curr.next;  
 curr.next = prev;  
 prev=curr;  
 curr=nextn;  
 }  
 nextn=head.next;  
 head.next = tail;  
 tail = nextn;  
 }  
 public void display () {  
 if (head == null) {  
 System.*out*.println("empty list");  
 return;  
 }  
 if (head==tail) {  
 System.*out*.println(head.val);  
 return;  
 }  
 Node curr = head;  
  
 while (curr != tail) {  
 System.*out*.print(curr.val + "-->");  
 curr = curr.next;  
 }  
 System.*out*.println(tail.val+"-->"+head.val);  
 System.*out*.println(size);  
 }  
  
}  
  
  
  
  
 public static void main(String[] args) {  
 CLL list = new CLL();  
 list.addFirst(6);  
 list.addFirst(5);  
 list.addFirst(4);  
 list.addFirst(3);  
 list.addLast(7);  
 list.addFirst(2);  
 list.addFirst(1);  
// list.display();  
 list.insert(4,11);  
  
// list.delFirst();  
// list.display();  
  
 list.delete(4);  
 list.rev();  
 list.delLast();  
 list.display();  
  
  
 }