/\*\*

\*/

public class CircularLinkedList<E> {

private Node<E> tail=null;

private int size=0;

public CircularLinkedList(){

}

public boolean isEmpty(){

return size==0;

}

public int size(){

return size;

}

public E first(){

if (isEmpty())return null;

return tail.getNext().getElement();

}

public E last()

{

if (isEmpty())return null;

return tail.getElement();

}

public void addFirst (E element){

if (size()==0)

{

tail=new Node <E> (element,null);

tail.setNext(tail);

}

else {

Node<E> n=new Node<E>(element,tail.getNext());

tail.setNext(n);

}

size++;

}

public void addLast(E element){

addFirst(element);

tail=tail.getNext();

}

public E removeFirst()

{

if (isEmpty())return null;

Node<E> x=tail.getNext();

if (x==tail)

tail=null;

else

tail.setNext(x.getNext());

size--;

return x.getElement();

}

public void rotate (){

if (tail!=null)

tail=tail.getNext();

}

public boolean equals (Object o){

if (o==null) return false;

if (getClass()!=o.getClass())return false;

CircularLinkedList other=(CircularLinkedList)o;

if (size!=other.size)return false;

CircularLinkedList.Node walkA=tail;

CircularLinkedList.Node walkB=other.tail;

while (walkA!=null){

if (!walkA.getElement().equals(walkB.getElement())) return false;

walkA=walkA.getNext();

walkB=walkB.getNext();

return true;

}

return true;

}

private static class Node<E>

{

E element;

Node<E> next;

public Node(E element, Node<E> next) {

this.element = element;

this.next = next;

}

public E getElement() {

return element;

}

public void setElement(E element)

{

this.element = element;

}

public Node<E> getNext()

{

return next;

}

public void setNext(Node<E> next)

{

this.next = next;

}

}

}