```
// DO NOT MODIFY THIS FILE
// It was generated by Hyper/J(tm) for debugging purposes only
// To modify the program, edit the source files noted in comments below

package container;

// Stub code generated by Hyper/J(tm). Use <step into> to reach actual methods.
/**
 * Deque.java
 * Classe Deque - um Deque implementado com lista duplamente encadeada
 * 18 julho de 2003
 * SCE - PGCC - UFSC
 * @author Augusto César
 */

//package container;


public class Deque {
        private DLNode header, trailer; // referencia para o nó do inicio
        private int size;       // numero de elementos

        //Construtor
        public Deque(){
                header = new DLNode();
                trailer = new DLNode();
                header.setNext(trailer);        // faz header apontar para trailer
                trailer.setPrev(header);        // faz trailer apontar para header
                size = 0;
        }

        public int size(){
                return size;
        }

        public boolean isEmpty(){
                if (header==null)
                        return true;
                return false;
        }

        public Object first(){
                if(isEmpty())
                        throw new RuntimeException("Deque está vazio.");
                return header.getNext().getElement();
        }

        public Object last(){
                if(isEmpty())
                        throw new RuntimeException("Deque está vazio.");
                return trailer.getPrev().getElement();
        }


        public void insertFirst( Object elem){
                DLNode second = header.getNext();
                DLNode first = new DLNode(elem,header, second);
                second.setPrev(first);
                header.setNext(first);
                size++;
        }

        public void insertLast( Object elem){
                DLNode second = trailer.getPrev();
                DLNode first = new DLNode(elem,header, second);
                second.setPrev(first);
                header.setNext(first);
                size++;
        }
}

/**
 * Deque.java
 * Classe Deque - um Deque implementado com lista duplamente encadeada
 * 18 agosto de 2003
 * SCE - PGCC - UFSC
 * @author Augusto César
 */
```

```
//package container.remocao;

import container.*;

public abstract class Deque {
	private DLNode header, trailer; // referencia para o nó do inicio
	private int size;       // numero de elementos

	//Construtor
	public Deque(){
		header = new DLNode();
		trailer = new DLNode();
		header.setNext(trailer);        // faz header apontar para trailer
		trailer.setPrev(header);        // faz trailer apontar para header
		size = 0;
	}
	//tamanho do Deque
	public  int size(){
		return size;
	}

	// Verifica se o Deque está vázio.
	public abstract boolean isEmpty();

	//Retorna o primeiro objeto
	public Object first(){
		if(isEmpty())
			throw new RuntimeException("Deque está vazio.");
		return header.getNext().getElement();
	}

	public Object last(){
		if(isEmpty())
			throw new RuntimeException("Deque está vazio.");
		return trailer.getPrev().getElement();
	}

	public Object removeFirst(){
		if(isEmpty())
			throw new RuntimeException("Deque está vazio.");
		DLNode first = header.getNext();
		Object o = first.getElement();
		DLNode secondToLast = first.getPrev();
		trailer.setPrev(secondToLast);
		secondToLast.setNext(trailer);
		size--;
		return o;
	}


	public Object removeLast() throws RuntimeException{
		if(isEmpty())
			throw new RuntimeException("Deque está vazio.");
		DLNode last = trailer.getPrev();
		Object o = last.getElement();
		DLNode secondToLast = last.getPrev();
		trailer.setPrev(secondToLast);
		secondToLast.setNext(trailer);
		size--;
		return o;
	}
}
```