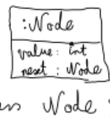
Hallo Konstantin!

1	3	4	-1	-3

```
public static int[] addElement(int[] a, int e){
    int[] r = new int[a.length + 1];
    for(int i=0; i<a.length; i++){
        r[i] = a[i];
    }
    r[r.length - 1] = e;
    return r;
}
```

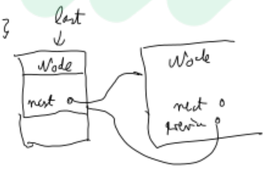



```
class List {
    Node first;
    private int size = 0;
public void addFirst(int e){
    Node n = new Node();
    n.value = e;
    n.next = this.first;
    this.first.previous = n;
    this.first = n;
    ++size;
}
```

```
public class Node {
    int value;
    Node next;
    Node previous;
```

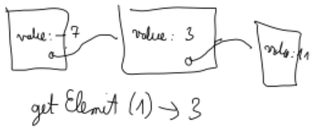

```
public void removeFirst(){
    if(this.first == null) return;
    this.first = this.first.next;
    this.first.previous = null; --size;
}
public int size(){
    int s = 0;
    Node n = this.first;
    while(n != null){
        n = n.next;
        s++;
    }
    return s;
}
public boolean contains(int e){
    Node n = this.first;
    while(n != null){
        if(n.value == e) return true;
        n = n.next;
    }
    return false;
}
public void addLast(int e){
    Node last = getLast();

    Node n = new Node();
    n.value = e;
    if(last == null){this.first = n; return;}
    last.next = n;
    n.previous = last;
    ++size;
```




```
}
private Node getLast(){
    Node n = this.first;
    while(n != null && n.next != null)
        n = n.next;
    return n;
}
public void removeLast(){
    Node last = getLast();
    if(last == null) return;
    if(last.previous == null){
        this.first = null;
        size = 0; return;
    }
    last.previous.next = null;
    size--;
}
public void addElementAt(int i, int e){

}
public int getElement(int i){
    Node n = this.first; int k=0;
    while(n != null && k < i){
        n = n.next;
        k++;
    }
    return n.value;
}
```



getElement(1) → 3