**Палиндром - Java Задача 1 (0 / 0)**

Дадена е двојно поврзана листа со N јазли каде секој јазел содржи по еден карактер (буква). Да се провери дали двојно поврзаната листа е палиндром: односно ако ја изминете од почеток до крај и од крај до почеток, дали ќе добиете ист збор. Во првиот ред од влезот даден е бројот на јазли во листата N, а во вториот ред се дадени броевите. На излез треба да се испечати 1 ако листата е палиндром, -1 ако не е.

Име на класата: PalindromeDLL

Вашето решение:

Програмски јазик: Java

import java.util.Scanner;

class DLLNode<E> {

protected E element;

protected DLLNode<E> pred, succ;

public DLLNode(E elem, DLLNode<E> pred, DLLNode<E> succ) {

this.element = elem;

this.pred = pred;

this.succ = succ;

}

@Override

public String toString() {

return "<-" + element.toString() + "->";

}

}

class DLL<E> {

private DLLNode<E> first, last;

public DLL() {

// Construct an empty SLL

this.first = null;

this.last = null;

}

public void deleteList() {

first = null;

last = null;

}

public int length() {

int ret;

if (first != null) {

DLLNode<E> tmp = first;

ret = 1;

while (tmp.succ != null) {

tmp = tmp.succ;

ret++;

}

return ret;

} else

return 0;

}

public void insertFirst(E o) {

DLLNode<E> ins = new DLLNode<E>(o, null, first);

if (first == null)

last = ins;

else

first.pred = ins;

first = ins;

}

public void insertLast(E o) {

if (first == null)

insertFirst(o);

else {

DLLNode<E> ins = new DLLNode<E>(o, last, null);

last.succ = ins;

last = ins;

}

}

public void insertAfter(E o, DLLNode<E> after) {

if (after == last) {

insertLast(o);

return;

}

DLLNode<E> ins = new DLLNode<E>(o, after, after.succ);

after.succ.pred = ins;

after.succ = ins;

}

public void insertBefore(E o, DLLNode<E> before) {

if (before == first) {

insertFirst(o);

return;

}

DLLNode<E> ins = new DLLNode<E>(o, before.pred, before);

before.pred.succ = ins;

before.pred = ins;

}

public E deleteFirst() {

if (first != null) {

DLLNode<E> tmp = first;

first = first.succ;

if (first != null) first.pred = null;

if (first == null)

last = null;

return tmp.element;

} else

return null;

}

public E deleteLast() {

if (first != null) {

if (first.succ == null)

return deleteFirst();

else {

DLLNode<E> tmp = last;

last = last.pred;

last.succ = null;

return tmp.element;

}

}

// else throw Exception

return null;

}

@Override

public String toString() {

String ret = new String();

if (first != null) {

DLLNode<E> tmp = first;

ret += tmp + "<->";

while (tmp.succ != null) {

tmp = tmp.succ;

ret += tmp + "<->";

}

} else

ret = "Prazna lista!!!";

return ret;

}

public DLLNode<E> getFirst() {

return first;

}

public DLLNode<E> getLast() {

return last;

}

}

public class PalindromeDLL {

public static int isItPalindrome(DLL<Integer> list){

//Vashiot kod tuka...

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int n = in.nextInt();

DLL<Integer> list = new DLL<Integer>();

for (int i = 0; i < n; i++) {

list.insertLast(in.nextInt());

}

in.close();

System.out.println(isItPalindrome(list));

}

}