36

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36

[1->2->3->4, 5->6->7->8, 9->10->11->-12, 13->14->15->16, 17->18->19->20, 21->22->23->24, 25->26->27, 28->29->30, 31->32->33, 34->35->36]

import java.util.Scanner;

class Element {

private int id;

public Element(int id) {

this.id = id;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

@Override

public String toString() {

return String.valueOf(id);

}

}

class SLLNode<E> {

protected E element;

protected SLLNode<E> succ;

public SLLNode(E elem, SLLNode<E> succ) {

this.element = elem;

this.succ = succ;

}

@Override

public String toString() {

return element.toString();

}

}

class SLL<E> {

private SLLNode<E> first;

public SLL() {

this.first = null;

}

public void deleteList() {

first = null;

}

public int length() {

int ret;

if (first != null) {

SLLNode<E> tmp = first;

ret = 1;

while (tmp.succ != null) {

tmp = tmp.succ;

ret++;

}

return ret;

} else

return 0;

}

@Override

public String toString() {

String ret = new String();

if (first != null) {

SLLNode<E> tmp = first;

ret += tmp;

while (tmp.succ != null) {

tmp = tmp.succ;

ret += " -> " + tmp;

}

} else

ret = "Prazna lista!!!";

return ret;

}

public void insertFirst(E o) {

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

first = ins;

}

public void insertAfter(E o, SLLNode<E> node) {

if (node != null) {

SLLNode<E> ins = new SLLNode<E>(o, node.succ);

node.succ = ins;

} else {

System.out.println("Dadenot jazol e null");

}

}

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

if (first != null) {

SLLNode<E> tmp = first;

if (first == before) {

this.insertFirst(o);

return;

}

while (tmp.succ != before)

tmp = tmp.succ;

if (tmp.succ == before) {

SLLNode<E> ins = new SLLNode<E>(o, before);

tmp.succ = ins;

} else {

System.out.println("Elementot ne postoi vo listata");

}

} else {

System.out.println("Listata e prazna");

}

}

public void insertLast(E o) {

if (first != null) {

SLLNode<E> tmp = first;

while (tmp.succ != null)

tmp = tmp.succ;

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

tmp.succ = ins;

} else {

insertFirst(o);

}

}

public E deleteFirst() {

if (first != null) {

SLLNode<E> tmp = first;

first = first.succ;

return tmp.element;

} else {

System.out.println("Listata e prazna");

return null;

}

}

public E delete(SLLNode<E> node) {

if (first != null) {

SLLNode<E> tmp = first;

if (first == node) {

return this.deleteFirst();

}

while (tmp.succ != node && tmp.succ.succ != null)

tmp = tmp.succ;

if (tmp.succ == node) {

tmp.succ = tmp.succ.succ;

return node.element;

} else {

System.out.println("Elementot ne postoi vo listata");

return null;

}

} else {

System.out.println("Listata e prazna");

return null;

}

}

public SLLNode<E> getFirst() {

return first;

}

public void setFirst(SLLNode<E> node){

first = node;

}

public SLLNode<E> find(E o) {

if (first != null) {

SLLNode<E> tmp = first;

while (tmp.element != o && tmp.succ != null)

tmp = tmp.succ;

if (tmp.element == o) {

return tmp;

} else {

System.out.println("Elementot ne postoi vo listata");

}

} else {

System.out.println("Listata e prazna");

}

return first;

}

}

public class Transformation {

private static void listTransform(SLL<Element> original) {

//TODO: Your code

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int num = Integer.parseInt(scanner.nextLine());

SLL<Element> list = new SLL<Element>();

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

int n = scanner.nextInt();

list.insertLast(new Element(n));

}

listTransform(list);

}

}