Ascending digits

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

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

int n=sc.nextInt();

int max=Integer.MIN\_VALUE;

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

int num=sc.nextInt();

list.add(num);

max=Math.max(String.valueOf(num).length(),max);

}

LinkedList<Integer> res=new LinkedList<>();

for (int i=1; i<=max; i++){

for (int v:list){

if (String.valueOf(v).length()==i){

res.add(v);

}

}

}

for (int i:res){

System.out.print(i+" ");

}

}

}

Jumping friend

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

int num=sc.nextInt();

list.add(num);

}

int p1=0, p2=0;

while (p2 < n-1){

p1++;

p2=Math.min(p2+3, n-1);

}

System.out.print(list.get(p1));

}

}

Linked list jumps

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

list.add(sc.nextInt());

}

int sum=0;

int cur=0;

while (cur<n){

sum+=list.get(cur);

int max=list.get(cur);

int next=cur;

for (int i=1; i<=5 && cur+i<n; i++){

int temp=list.get(cur+i);

if (temp>max){

max=temp;

next=cur+i;

}

}

if (next==cur){

break;

}

cur=next;

}

System.out.print(sum);

}

}

Insert and delete

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

sc.nextLine();

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

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

String s=sc.nextLine();

String arr[]=s.split(" ");

String action=arr[0];

int value=Integer.parseInt(arr[1]);

if (action.equals("insert")){

insert(list,value);

}

else if (action.equals("delete")){

delete(list,value);

}

}

for (int i:list){

System.out.print(i+" ");

}}

static void insert(LinkedList<Integer> list, int value){

int size=list.size();

int mid=size/2;

if (size==0){

list.add(value);

}

else {

list.add(mid,value);

}

}

static void delete(LinkedList<Integer> list, int value){

list.removeIf(num -> num==value);

}

}

Form the string

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

sc.nextLine();

LinkedList<String>list=new LinkedList<>();

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

list.add(sc.next());

}

sc.nextLine();

String target=sc.nextLine();

LinkedList<String> ans=new LinkedList<>();

StringBuilder cur =new StringBuilder();

for (String i:list){

if (cur.length()<target.length() && target.startsWith(cur.toString()+i)){

ans.add(i);

cur.append(i);

if (cur.toString().equals(target)){

break;

}

}

}

if (cur.toString().equals(target)){

System.out.print(String.join(" ",ans));

}

else{

System.out.print(-1);

}

}

}

Two-iteration pair swap

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

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

int n=sc.nextInt();

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

list.add(sc.nextInt());

}

for (int i=0; i<n-1; i++){

if (list.get(i)%2==0 && list.get(i+1)%2!=0){

int temp=list.get(i);

list.set(i,list.get(i+1));

list.set(i+1,temp);

}

}

for (int i=n-1; i>0; i--){

if (list.get(i)%2!=0 && list.get(i-1)%2==0){

int temp=list.get(i);

list.set(i,list.get(i-1));

list.set(i-1,temp);

}

}

for (int i:list){

System.out.print(i+" ");

}

}

}

Fix rotated sorted linked list

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

list.add(sc.nextInt());

}

Collections.sort(list,Collections.reverseOrder());

for (int i:list){

System.out.print(i+" ");

}

}

}

Delete all evens or odds

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

list.add(sc.nextInt());

}

Iterator<Integer>k=list.iterator();

boolean del=list.getFirst()%2==0;

while (k.hasNext()){

int c=k.next();

if (del && c%2==0){

k.remove();

}

else if (!del && c%2!=0){

k.remove();

}

}

for (int i:list){

System.out.print(i+" ");

}

}

}

Node switching

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

list.add(sc.nextInt());

}

swap(list);

rev(list);

}

static void swap(LinkedList<Integer> list){

for (int i=0; i<list.size(); i+=3){

if (i+2<list.size()){

Integer first=list.get(i);

Integer second=list.get(i+1);

Integer third=list.get(i+2);

list.remove(i+1);

list.set(i,third);

list.set(i+1,first);

i-=1;

}

}

}

static void rev(LinkedList<Integer> list){

for (int i=list.size()-1; i>=0; i--){

System.out.print(list.get(i)+" ");

}

}

}

Add walls

import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc=new Scanner (System.in);

int n=sc.nextInt();

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

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

list.add(sc.nextInt());

}

int k=sc.nextInt();

int sum=0;

for (int i=0; i<list.size(); i++){

sum+=list.get(i);

if (sum>k){

list.add(i,0);

sum=list.get(i+1);

i++;

}

}

for (int i:list){

System.out.print(i+" ");

}

}

}