import static java.lang.Thread.sleep;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class Lamport {

public static void main(String[] args) throws InterruptedException {

Scanner sc = new Scanner(System.in);

System.out.println("Enter number of Processes: ");

int n = sc.nextInt();

List<Integer> sort = new ArrayList<Integer>();

System.out.println("Enter Timestamp of processes who need critical region(100 for NI):");

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

System.out.print("P" + (i + 1) + ": ");

sort.add(sc.nextInt());

}

System.out.println("");

int iterator = 0;

while (Collections.min(sort) != 100) {

int min = Collections.min(sort);

int min\_index = sort.indexOf(Collections.min(sort));

if (iterator == 0) {

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

if (sort.get(i) != 100) { //process who are interested in entering CR

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

if (i != j) { //so that p1 does not send req to p1

System.out.println("P" + (i + 1) + " -> REQ -> P" + (j + 1));

}

}

}

}

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

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

if (j != min\_index && i != j && sort.get(i) != 100) {

System.out.println(" P" + (i + 1) + " <- OK <- P" + (j + 1));

}

}

}

}

System.out.println("P" + (min\_index + 1) + " gets access to CR");

sleep(5000);

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

if (i != min\_index) {

System.out.println("P" + (min\_index + 1) + " -> REL -> P" + (i + 1));

}

}

System.out.println("");

sort.set(min\_index, 100);

iterator += 1;

}

}

}

