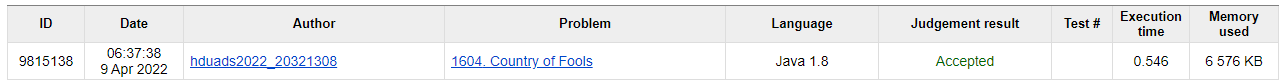
Laboratory work #3

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1604

Screenshot from Timus:



Explanation of algorithm:

To make the adjacent signs as different as possible, when the type of remaining signs is greater than 2, we need to find the most and the second most signs, and then place them; When the remaining sign type is 1, place them all.

Computational complexity of algorithm:

F(N) = N + N + N^2 = 2N + N^2

T(N) = O(N^2)

Source code:

import java.util.Scanner;

public class App {

public static int Count(int[] arr) {

int count = 0;

for (int i = 0; i < arr.length; i++) {

if (arr[i] != 0)

count++;

}

return count;

}

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

Scanner scan = new Scanner(System.in);

int k = scan.nextInt();

int[] n = new int[k];

String ans = "";

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

n[i] = scan.nextInt();

}

int count = Count(n);

while (count > 0) {

int max = 0;

int index\_max = 0;

int sec = 0;

int index\_sec = 0;

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

if (n[i] > max && n[i] > 0) {

max = n[i];

index\_max = i;

}

}

if (count > 1) {

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

if (n[i] > sec && index\_max != i && n[i] > 0) {

sec = n[i];

index\_sec = i;

}

}

n[index\_max]--;

n[index\_sec]--;

int a = index\_max + 1;

int b = index\_sec + 1;

ans += a + " " + b + " ";

} else if (count == 1) {

n[index\_max]--;

int a = index\_max + 1;

ans += a + " ";

}

count = Count(n);

}

System.out.println(ans);

scan.close();

}

}