KỲ THI CHỌN HỌC SINH GIỎI CẤP TỈNH THCS NĂM 2023

TỈNH QUẢNG NINH

MÔN TIN HỌC – BẢNG A

Ngày thi 14/03/2023 – Thời gian 150 phút

TỔNG QUAN ĐỀ THI

Bài	Tên bài	File CT	File input	File output	Điểm
1	Màn hình	DISP.*	DISP.INP	DISP.OUT	6
2	Dãy số tăng	INCR.*	INCR.INP	INCR.OUT	6
3	Bánh mì và bánh rán	DONU.*	DONU.INP	DONU.OUT	5
4	Giờ học giáo dục thể chất	PHYS.*	PHYS.INP	PHYS.OUT	3

(là pas, cpp hoặc py)*

Bài 1. DISP Màn hình

Một công ty lớn đã quyết định đưa ra một loại màn hình có đúng n điểm ảnh được xếp thành các hàng và các cột.

Nhiệm vụ của bạn là xác định số hàng điểm ảnh a và số cột điểm ảnh b sao cho:

- Có đúng n điểm ảnh trên màn hình, tức là a x b = n
- Số hàng điểm ảnh không vượt quá số cột điểm ảnh, tức là $a \leq b$
- Sự khác biệt b – a càng nhỏ càng tốt.

Input: gồm một dòng chứa số nguyên n ($1 \leq n \leq 10^9$)

Ouput: ghi hai số nguyên tương ứng là số hàng và số cột điểm ảnh cần tìm của màn hình.

Ràng buộc:

- 30% test thỏa $1 \leq n \leq 10^3$
- 30% test thowowra $1 \leq n \leq 10^7$.
- 40% test không ràng buộc gì thêm

Input	Output
8	2 4
25	5 5

giải

vì b – a nhỏ nhất có thể nên ta duyệt a từ int(sqrt(n)) trở về 1.

nếu gặp giá trị a đầu tiên thòa n chia hết cho a thì a và b = n/a là cặp cần tìm

code c++

```
#include<bits/stdc++.h>
using namespace std;

main() {
     ios_base::sync_with_stdio(false);
    cin.tie(0);
    int n;
    cin>>n;
    for(int a=int(sqrt(n));a>0;a--){
        if(n%a==0){
```

```
            cout<<a<<" "<<n/a;
            break;
        }
    }
}
```

Bài 2. INCR Dãy số tăng

Dãy số $a_1, a_2, \ldots, a_n$ được gọi là tăng nếu $a_1 < a_2 < \ldots < a_n$.

Bạn được cho một dãy số $b_1, b_2, \ldots, b_n$ và một số nguyên dương d. Trong mỗi lần thao tác, bạn chọn một phần tử của dãy số và cộng thêm d vào nó. Số thao tác ít nhất là bao nhiêu để biến đổi dãy số đã cho trở thành dãy số tăng.

Input

- Dòng đầu tiên chứa hai số nguyên n và d ($1 \le n \le 10^5$; $1 \le d \le 10^9$).
- Dòng thứ hai chứa n số nguyên tương ứng là dãy số $b_1, b_2, \ldots, b_n$ ($1 \le bi \le 10^9$)

Output: ghi một số nguyên là số thao tác ít nhất để biến đổi dãy số đã cho trở thành dãy số tăng.

Ràng buộc:

30% test thỏa $1 \le n, d, b_i \le 10^2$.

30% test thỏa $1 \le n \le 10^3$ và $1 \le d, b_i \le 10^6$.

40% test không có ràng buộc gì thêm.

Input	Output	Giải thích
4 2 1 3 3 2	3	dãy b là 1, 3, 3, 2 và d = 2. Số thao tác ít nhất để biến đổi dãy số trở thành dãy số tăng là 3 và một trong các cách thực hiện như sau:\ Thao tác 1: cộng thêm d vào phần tử thứ ba, dãy trở thành 1, 3, 5, 2 Thao tác 2: cộng thêm d vào phần từ thứ tư, dãy trở thành 1, 3, 5, 4 Thao tác 3: cộng thêm d vào phần từ thứ tư, dãy trở thành 1, 3, 5, 6 và là dãy số tăng.

giải, ktlt, m1c

duyệt i từ 2 đến n

nếu a[i] ≤ a[i – 1] thì cần tác động

số thác tác cần thực hiện lên a[i] là x = (a[i-1] – a[i])/d + 1 để a[i] > a[i-1]

cập nhật a[i]

code c++

```
#include<bits/stdc++.h>
using namespace std;

long long a[1000000];

main() {
     ios_base::sync_with_stdio(false);
    cin.tie(0);
    long long n, d;
    cin>>n>>d;
    long long x,res=0;
```

```
    for(int i=1;i<=n;i++) cin>>a[i];
    for(int i=2;i<=n;i++){
        if(a[i]<=a[i-1]){
            x = a[i-1]-a[i];
            res = res + x/d+1;
            a[i] = a[i] + d*(x/d+1);
        }
    }
    cout<<res;
}
```

Bài 3. DONU Bánh mì và bánh rán

Mẹ của An đã lên kế hoạch ăn sáng bằn gbasnh mì hoặc bánh rán cho An trong n ngày (được đánh số từ 1 đến n). Mẹ của An viết một xâu s độ dài n, trong đó kí tự thứ i ($1 \le i \le n$) là '0' hoặc '1' biểu thị ngày thứ i sẽ ăn bánh mì hoặc bánh rán tương ứng.

An thích ăn bánh rán hơn bánh mì, nên anh ta muốn chọn một đoạn gồm k ký tự liên tiếp trong xâu s và thay đổi mỗi ký tự '0' trong đoàn thành '1'. Gọi ***time*** là số ngày liên tiếp dài nhất mà Anh ăn bánh rán. Bạn hãy giúp An tìm giá trị ***time*** lớn nhất mà anh ta có thể đạt được bằng cách chọn một đoạn hợp lý.

Input

- Dòng đầu chứa hai số nguyên n và k ($1 \le k \le n \le 10^6$).
- Dòng thứ hai chứa xâu s đội dài n, chỉ gồm các ký tự '0' và '1'.

Output: ghi một số nguyên là giá trị time lớn nhất.

Ràng buộc:

- 30% test thỏa mãn $1 \le k \le n \le 10^2$
- 30% test thỏa $1 \le k \le n \le 10^3$.
- 40% test không có ràng buộc nào thêm.

Input	Output	Giải thích
13 2 0101110000101	5	An cần chọn đoạn ký tự từ thứ 2 đến thứ 3 là "10", sau đó thay đổi ký tự thứ 3 trong s thành '1' và time là 5 ngày: từ thứ 2 đến thứ 6
6 3 100001	4	An cần chọn đoạn ký tự từ thứ 2 đến thứ 4 là "000", sau đó thay đổi các ký tự '0' thành '1' và time là 4 ngày: từ thứ 1 đến thứ 4

giải, mảng tiền tố, hậu tố

với mỗi vị trí i ta tính trước

f[i] = chiều dài dãy con kí tự '1' liên tiếp kết thúc tại vị trí i tính từ trái sang.

Vd: f[6] = 3, f[7] = 0

g[i] = chiều dài dãy con kí tự '1' liên tiếp kết thúc tại vị trí i tính từ phải sang.

Vd: g[11] = 1, g[10] = 0

Duyệt qua từng vị trí i, ta có đoạn con '1' dài nhất khi lấy vị trí là chuẩn là

f[i – 1] + k + g[i + k]

ví dụ: lấy vị trí i = 3 làm chuẩn

13 2
0101110000101
F[i – 1] = 1 (màu đỏ), k = 2 (màu xanh), g[i + k] = 2 (màu vàng), ta được đoạn con ‘1’ có chiều dài 5
Ta lấy max của tất cả các vị trí

code c++

```
#include<bits/stdc++.h>
using namespace std;

const int A = 1000005;
int n, k;
string s;
int f[A], g[A];

void read() {
    cin >> n >> k >> s;
}

void solve() {
    s = ' ' + s;
    for(int i = 1; i <= n; ++i)
        if(s[i] == '1')
            f[i] = f[i - 1] + 1;
    for(int i = n; i >= 1; --i)
        if(s[i] == '1')
            g[i] = g[i + 1] + 1;
    int res = 0;
    for(int i = 1; i <= n - k + 1; ++i)
        res = max(res, f[i - 1] + k + g[i + k]);
    cout << res;
}

main() {
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    read();
    solve();
}
```

Bài 4. PHYS Giờ học giáo dục thể chất

Trước giờ học giáo dục thể chất, một lớp gồm n học sinh xếp thành một hàng. Tất cả học sinh trong lớp đều có chiều cao khác nhau. Vị trí thứ i (i = 1, 2, …, n) tính từ đầu bên trái của hàng là học sinh có chiều cao p_i ($1 \le p_i \le n$).

Khi bắt đầu giờ học, thầy giáo có thể thay đổi thứ tự học sinh trong hàng. Để làm điều này, thầy giáo có thể thực hiện thao tác sau **đúng** một lần: chọn một đoạn từ vị trí l đến vị trí r ($1 \le l \le r \le n$) và sắp xếp các học sinh trong đoạn này theo chiều cao tăng dần từ trái qua phải. Ví dụ n = 5,

ban đầu học sinh theo thứ tự có chiều cao 5, 2, 4, 1, 3 và thầy giáo chọn l = 1, r = 4 thì sau khi sắp xếp học sinh sẽ theo thứ tự có chiều cao là 1, 2, 4, 5, 3.

Sử dụng thao tác này, thầy giáo có thể sắp xếp để hai học sinh nào đó cách xa nhau nhất có thể. Khoảng cách giữa hai học sinh bằng sự chênh lệch giữa các vị trí mà hai học sinh đứng. Với mỗi cặp học sinh, thầy giáo tính khoảng cách lớn nhất giữa hai học sinh này sau khi thực hiện đúng 1 thao tác nói trên. Bạn hãy giúp thầy giáo tìm tổng của các giá trị này.

Cụ thể hơn, hãy xét hai học sinh ban đầu ở vị trí i và j ($1 \le i < j \le n$). Gọi d(i, j) là khoảng cách lớn nhất giữa hai học sinh đó mà thầy giáo có thể đạt được bằng cách chọn một đoạn và sắp xếp. Bạn cần tính tổng tất cả các giá trị d(i, j) với mọi i, j thỏa mãn $1 \le i < j \le n$

Input

- Dòng đầu tiên chứa một số nguyên n là số học sinh trong lớp ($2 \le n \le 3000$)
- Dòng thứ hai chứa n số nguyên $p_1, p_2, \ldots, p_n$ là chiều cao của từng học sinh trong hàng ($1 \le p_i \le n$). Dữ liệu đảm bảo rằng tất cả các số p_i đều phân biệt.

Output: ghi một số nguyên là câu trả lời cho bài toán.

Ràng buộc:

- 20% test có $n \le 10$,
- 20% test khác có $n \le 50$,
- 20% test khác có $n \le 100$,
- 20% test khác có $n \le 600$,
- 20% test còn lại không có ràng buộc gì thêm.

Input	Output	Giải thích
5 5 2 4 1 3	35	Câu trả lời là tổng các số sau: d(1, 2) = 3, d(1, 3) = 4, d(1, 4) = 4, d(1, 5) = 4, d(2, 3) = 3, d(2, 4) = 3, d(2, 5) = 4, d(3, 4) = 3, d(3, 5) = 3, d(4, 5) = 4. Ví dụ với hai học sinh ban đầu đứng ở vị trí 4 và 5, có chiều cao lần lượt là 1 và 3, thầy giáo có thể chọn đoạn với l = 1 và r = 4. Khi đó dãy học sinh sẽ thay đổi như sau: 5, 2, 4, 1, 3 → 1, 2, 4, 5, 3 (Đoạn đã chọn được gạch dưới) và khoảng cách giữa hai học sinh này là 4.
10 2 1 6 8 3 5 9 10 7 4	256	
2 2 1	1	

giải

Ta cần duyệt qua tất cả các vị trí i và j để tính d(i, j).

Với hai vị trí i và j, ta có 3 trường hợp sort một đoạn l đến r sao cho:

- i và j nằm trong đoạn l đến r.

Để tối ưu ta sẽ sort hết đoạn thì sẽ có càng nhiều số nằm giữa 2 số p[i] và p[j]

ví dụ:

2 **1** 6 8 3 5 **9** 10 7 4 → **1** 2 3 4 5 6 7 8 **9** 10

tính d(2, 7), ta sẽ sort hết dãy a[]. Thực chất ta có thể tính ngay d(2, 7) = abs(p[2] - p[7]) = abs(1 – 9) = 8

- i nằm trong đoạn L đến R còn j thì không. vì i < j nên để i xa j nhất thì ta phải sort đoạn 1 đến i để đảm bảo có nhiều số lớn hơn p[i] nhất

ví dụ:

2 1 6 8 **3** 5 9 10 7 **4** → 1 2 **3** 6 8 5 9 10 7 **4**

tính d(5, 10), ta sẽ sort đoạn a[1..5]. Thực chất ta có thể tính ngay d(5, 10) = 10 – f[5], với f[5] là số phần tử nhỏ hơn hoặc bằng p[5] trong đoạn a[1..5]

- j nằm trong đoạn L đến R còn i thì không. để i xa j nhất thì phải sort đoạn từ j đến n để đảm bảo có nhiều số bé hơn p[j] nhất

ví dụ:

2 1 6 8 3 5 **9** 10 7 4 → **2** 1 6 8 3 5 4 7 **9** 10

tính d(1, 7), ta sẽ sort đoạn a[7..10]. Thực chất ta có thể tính ngay d(1, 7) =g[7] - 1, với g[7] là (n + 1) - số phần tử lớn hơn hoặc bằng p[7] trong đoạn a[7..10]

Kết quả của d(i, j) là max của 3 trường hợp. Tổng các d(i, j) là ra kết quả.

code c++

```
#include<bits/stdc++.h>
using namespace std;

void fre() {
    ios_base::sync_with_stdio(false);
    cin.tie(0);
}

const int A = 3005;

int n;
long long p[A],f[A],g[A];

void read() {
    cin >> n;
    for(int i = 1; i <= n; ++i)
        cin >> p[i];
}

void solve() {
    for(int i = 1; i <= n; ++i) {
        for(int j = 1; j <= i; ++j)
            if(p[j] <= p[i])
                ++f[i];  //so phan tu doan [1,i] <= p[i]
        g[i] = n + 1;
        for(int j = i; j <= n; ++j)
            if(p[j] >= p[i])
                --g[i];
    }
    long long res = 0;
    for(int i = 1; i < n; ++i)
        for(int j = i + 1; j <= n; ++j)
```

```
            res += max({abs(p[i] - p[j]), j - f[i], g[j] - i});
    cout << res;
}

main() {
    fre();
    read();
    solve();
}
```