liouzhou_101's blog

Editorial of Codeforces Round #700

By liouzhou_101, history, 72 minutes ago,

1480A - Yet Another String Game

Tutorial

1480B - The Great Hero

Tutorial

1479A - Searching Local Minimum

Tutorial

1479B1 - Painting the Array I

Tutorial

[problem:1479B1]

Formally, for every sequence $a_1, a_2, \ldots, a_n$, and we assume that $a_1, a_2, \ldots, a_n$ are positive integers, the number of segments in a is defined to be

$$seg(a) = \sum_{i=1}^{n} [a_{i-1} \neq a_i],$$

where $a_0 = 0$, and $[\text{Boolean expression}] = 1$ if the Boolean expression is true and 0 otherwise.

Let's restate the problem as

Problem. Given a sequence $a_1, a_2, \ldots, a_n$, divide it into two disjoint subsequences s and t such that $seg(s) + seg(t)$ is as large as possible.

Solution. We will construct two disjoint subsequences by scanning through the sequence $a_1, a_2, \ldots, a_n$.

Initial setting: s and t are two empty sequences, and $a_1, a_2, \ldots, a_n$ remains not scanned.

Move on: Suppose the last elements of s and t are x and y, respectively, and $x = 0$ (resp. $y = 0$) if s (resp. t) is empty. Let z be the current element scanning through $a_1, a_2, \ldots, a_n$. Our greedy strategy is described in two cases:

- Greedy Strategy I: If z equals to one of x and y, then assign z to the opposite subsequence. That is, if $z = x$, then append z after y; and if $z = y$, then append z after x. In particular, if z equals to both x and y, the assignment could be arbitrary.
- Greedy Strategy II: If z differs from both x and y, then append z after the one with the nearest next same value. That is, let $next(x)$ denote the next position where x appears in $a_1, a_2, \ldots, a_n$ after z, then append z after x if $next(x) < next(y)$, and after y otherwise.

The greedy strategy is intuitive, and with this strategy, an $O(n)$ algorithm is immediately obtained. However, its proof turns out to be complicated. We append its proof for

→ Pay attention

Before contest
Codeforces Round #701 (Div. 2)
5 days

Like One person likes this. Sign Up to see what your friends like.

→ JacobianDet

Rating: 1426
Contribution: -5

- Settings
- Blog
- Favourites
- Teams
- Submissions
- Problemsetting
- Groups
- Talks
- Contests

JacobianDet

→ Top rated

#	User	Rating
1	tourist	3748
2	Benq	3550
3	maroonrk	3461
4	Um_nik	3437
5	ecnerwala	3386
6	Radewoosh	3352
7	jiangly	3328
8	scott_wu	3313
9	Petr	3309
10	boboniu	3289

Countries | Cities | Organizations View all →

→ Top contributors

#	User	Contrib.
1	1-gon	201
2	Errichto	198
3	rng_58	195
4	SecondThread	193
5	awoo	188
6	Um_nik	182
7	vovuh	180
8	Ashishgup	175
8	antontrygubO_o	175
10	-is-this-fft-	174

completeness.

An Intuitive Proof

Consider any optimal assignment $b_1, b_2, \ldots, b_n$, we will show that our strategy is not worse than it. Let $a[l \ldots r] = a_l, a_{l+1}, \ldots, a_r$ be the subarray of a.

Now suppose we are at some position p, where the optimal assignment conflicts with our strategy. We assume that $s = (a[1 \ldots p])^{(0)} = s'x$ ends with x, and $t = (a[1 \ldots p])^{(1)} = t'y$ ends with y, and $a_{p+1} = z$.

Greedy Strategy I: If b conflicts with Greedy Strategy I, then we must have $x \neq y$ and without loss of generality, we assume that $x = z$. Greedy Strategy I suggests we append z after y but b suggests we append z after x. Suppose b results in the two subarrays $\begin{matrix} s'xzs'' \\ t'yt'' \end{matrix}$, while there is indeed another optimal assignment that agrees with our strategy and results in $\begin{matrix} s'xt'' \\ t'yzs'' \end{matrix}$.

Greedy Strategy II: If b conflicts with Greedy Strategy II, then we must have x, y and z are distinct and without loss of generality, we assume that the next occurrence of x goes in front of that of y. Greedy Strategy II suggests we append z after x but b suggests we append z after y. Suppose b results in the two subarrays $\begin{matrix} s'xs'' \\ t'yzt'' \end{matrix}$. Consider two cases.

Case 1. If s'' does not start with y, then there is another optimal assignment that agrees with our strategy and results in $\begin{matrix} s'xzt'' \\ t'ys'' \end{matrix}$.

Case 2. If s'' starts with y, i.e. $s'' = ys_1$, then since the first occurrence of x is in front of that of y, we have that x must be in t'', and assume that $t'' = t_1xt_2$. The result of b is restated as $\begin{matrix} s'xys_1 \\ t'yzt_1xt_2 \end{matrix}$. We find that there is another optimal assignment that agrees with our strategy and results in $\begin{matrix} s'xzt_1ys_1 \\ t'yxt_2 \end{matrix}$ (Note that t_1 does not contain any x or y in it).

A Formal Proof

The number of alternations in a sequence a starting with x is defined to be

$$seg_x(a) = \sum_{i=1}^{n} [a_{i-1} \neq a_i],$$

where $a_0 = x$. We note that $seg_0(a) = seg(a)$.

Let $f_{x,y}(a)$ denote the maximal possible sum of numbers of alternations in the two disjoint subsequences s and t of a, i.e.

$$f_{x,y}(a) = \max_{s,t}\{seg_x(s) + seg_y(t)\},$$

where s and t ranges over all possible pairs of disjoint subsequences of a. It is obvious that the order of x and y does not matter, i.e. $f_{x,y}(a) = f_{y,x}(a)$. We note that our goal is to compute $f_{0,0}(a)$.

Let $next(x)$ denote the least index k such that $a_k = x$, i.e. $next(x) = \min\{k \in \mathbb{N} : a_k = x\}$. In case no such index k exists, $next(x)$ is defined to be ∞.

In fact, our problem can be solved by DP regardless of the time complexity.

Proposition 1 (Dynamic Programming). For $n \geq 1$ and every $x, y \in \mathbb{N}$,

$$f_{x,y}(a_1, a_2, \ldots, a_n) = \max\{f_{a_1,y}(a_2, \ldots, a_n) + [a_1 \neq x], f_{x,a_1}(a_2, \ldots, a_n) + [a_1 \neq y]\}$$

In particular, for empty sequence ϵ, we have $f_{x,y}(\epsilon) = 0$.

We can obtain some immediate properties of $f_{x,y}(a)$ by the above DP recurrence.

View all →

→ Favourite groups

#	Name
1	Code Marathon 6.0
2	Code Marathon 5.0
3	Codezilla
4	Codeforces Trainings Season 01 (2013)
5	1st Hunger Games
6	ACM-OI

View all →

→ Find user

Handle:

Find

→ Recent actions

Eurus7 → Round 700 — Thanx for strong pretests!

liouzhou_101 → Editorial of Codeforces Round #700

liouzhou_101 → Codeforces Round #700

Candidate_Master_2021 → Why do contest setter do this to us?

i_will_be_less_than_zero → IOI2021 Antarctica Team

Nezzar → Editorial of Codeforces Round #698 (Div.1, Div.2)

Mr_Tree_HLD_LCA_Master → Codeforces Vs Codechef Payments

Shanks → Codeforces Round #699 (Div. 2) Editorial

not-an-alt → TCS CodeVita Season 9 Qualifiers 2021 [CONTEST ENDED]

vlchen888 → Quora Programming Challenge 2021

Theo830 → Tester cheating in Codeforces Round #698!

blankSpace → Advice on Practice!!

Tlatoani → Codeforces Global Round 10

panipuri8 → Presenting to you the biggest cheat on Codeforces!

stefdasca → A quite interesting trick involving Disjoint Set Union

snapdragon3101 → CP Editor 5.0 : Submit to codeforces right from CP Editor

harkyn → So I'm a total noob, and I have a pretty hard time dealing with simple numerical logic.

chokudai → AtCoder Beginner Contest 191 Announcement

Um_nik → XXI OpenCup GP of Nizhny Novgorod

MikeMirzayanov → Rule about third-party code is changing

Elegia → IOI2021 China Team

Proposition 2. For every $x, y \in \mathbb{N}$, $f_{x,0}(a) \geq f_{x,y}(a) \geq f_{x,x}(a)$. Moreover, if $next(y) = \infty$, then $f_{x,0}(a) = f_{x,y}(a)$.

After some observations, we have

Proposition 3. For every $x, y, z \in \mathbb{N}$ and sequence a, $f_{z,x}(a) + 1 \geq f_{z,y}(a)$.

Proof: By induction on the length n of sequence a.

Basis. It is trivial for the case $n = 0$ since the left hand side is always 1 and the right hand side is always 0.

Induction. Suppose true for the case $n = k (k \geq 0)$, i.e.

$$f_{z,x}(a) + 1 \geq f_{z,y}(a)$$

holds for every sequence a of length k. Now consider a sequence $a_1, a_2, \ldots, a_{k+1}$ of length $k+1$.

Case 1. $x = y$. It is trivial that $f_{z,x}(a) + 1 \geq f_{z,x}(a)$.

Case 2. $z = x \neq y$. We should prove that $f_{x,x}(a) + 1 \geq f_{x,y}(a)$. By Proposition 1, we need to prove that

$$\begin{cases} f_{a_1,x}(a_2, \ldots, a_{k+1}) + [a_1 \neq x] + 1 \geq f_{a_1,y}(a_2, \ldots, a_{k+1}) + [a_1 \neq x], \\ f_{a_1,x}(a_2, \ldots, a_{k+1}) + [a_1 \neq x] + 1 \geq f_{x,a_1}(a_2, \ldots, a_{k+1}) + [a_1 \neq y]. \end{cases}$$

The second inequality is obvious. The first inequality becomes

$$f_{a_1,x}(a_2, \ldots, a_{k+1}) + 1 \geq f_{a_1,y}(a_2, \ldots, a_{k+1}),$$

which holds by induction.

Case 3. $x \neq y = z$. We should prove that $f_{x,y}(a) + 1 \geq f_{x,x}(a)$. By Proposition 1, we only need to prove that

$$f_{x,a_1}(a_2, \ldots, a_{k+1}) + [a_1 \neq y] + 1 \geq f_{a_1,x}(a_2, \ldots, a_{k+1}) + [a_1 \neq x],$$

which is obvious.

Case 4. $x \neq y, z \neq x$ and $z \neq y$. By Proposition 1, $f_{z,x}(a) + 1 \geq f_{z,y}(a)$ is equivalent to

$$\begin{aligned} &\max\{f_{a_1,x}(a_2, \ldots, a_{k+1}) + [a_1 \neq z], f_{z,a_1}(a_2, \ldots, a_{k+1}) + [a_1 \neq x]\} + 1 \\ &\geq \max\{f_{a_1,y}(a_2, \ldots, a_{k+1}) + [a_1 \neq z], f_{z,a_1}(a_2, \ldots, a_{k+1}) + [a_1 \neq y]\}. \end{aligned}$$

Case 4.1. $a_1 = z$. The left hand side becomes

$$\max\{f_{z,x}(a_2, \ldots, a_{k+1}), f_{z,z}(a_2, \ldots, a_{k+1}) + 1\} + 1 = f_{z,z}(a_2, \ldots, a_{k+1}) + 2$$

by induction that $f_{z,z}(a_2, \ldots, a_{k+1}) + 1 \geq f_{z,x}(a_2, \ldots, a_{k+1})$. The right hand side becomes

$$\max\{f_{z,y}(a_2, \ldots, a_{k+1}), f_{z,z}(a_2, \ldots, a_{k+1}) + 1\} = f_{z,z}(a_2, \ldots, a_{k+1}) + 1$$

by induction that $f_{z,z}(a_2, \ldots, a_{k+1}) + 1 \geq f_{z,y}(a_2, \ldots, a_{k+1})$. The inequality holds immediately.

Case 4.2. $a_1 = x$. The left hand side becomes

$$\max\{f_{x,x}(a_2, \ldots, a_{k+1}) + 1, f_{z,x}(a_2, \ldots, a_{k+1})\} + 1 = f_{x,x}(a_2, \ldots, a_{k+1}) + 2$$

by induction that $f_{x,x}(a_2, \ldots, a_{k+1}) + 1 \geq f_{z,x}(a_2, \ldots, a_{k+1})$. The right hand side becomes

$$\max\{f_{x,y}(a_2, \ldots, a_{k+1}) + 1, f_{z,x}(a_2, \ldots, a_{k+1}) + 1\}.$$

By induction that $f_{x,x}(a_2, \ldots, a_{k+1}) + 1 \geq f_{x,y}(a_2, \ldots, a_{k+1}) + 1$ and $f_{x,x}(a_2, \ldots, a_{k+1}) + 1 \geq f_{x,z}(a_2, \ldots, a_{k+1}) + 1$, the inequality holds.

Case 4.3. $a_1 = y$. The left hand side becomes

cpforfunCmnt → Bug In Edu Round 102 still not fixed!!!

Warawreh → Codeforces Round #699 (Div. 2)

pohzipohzi → Go: []byte(string(byteSlice)) != byteSlice?

Abhi3970 → My first contest ^ ^ memory which I now understand

Detailed →

$$\max\{f_{y,x}(a_2,\ldots,a_{k+1})+1, f_{z,y}(a_2,\ldots,a_{k+1})+1\}+1.$$

The right hand side becomes

$$\max\{f_{y,y}(a_2,\ldots,a_{k+1}), f_{z,y}(a_2,\ldots,a_{k+1})+1\} = f_{z,y}(a_2,\ldots,a_{k+1})+1$$

by induction that $f_{y,y}(a_2,\ldots,a_{k+1})+1 \geq f_{z,y}(a_2,\ldots,a_{k+1})+1$. The inequality immediately holds as $f_{z,y}(a_2,\ldots,a_{k+1})$ appears in both sides (and can be eliminated together).

Case 4.4. $a_1 \notin \{x,y,z\}$. The left hand side becomes

$$\max\{f_{a_1,x}(a_2,\ldots,a_{k+1})+1, f_{z,a_1}(a_2,\ldots,a_{k+1})+1\}+1.$$

The right hand side becomes

$$\max\{f_{a_1,y}(a_2,\ldots,a_{k+1})+1, f_{z,a_1}(a_2,\ldots,a_{k+1})+1\}.$$

By induction that $f_{a_1,x}(a_2,\ldots,a_{k+1})+1 \geq \max\{f_{a_1,y}(a_2,\ldots,a_{k+1}), f_{a_1,z}(a_2,\ldots,a_{k+1})\}$, the inequality holds.

The inequality holds for all cases. Therefore, the inequality holds for $n=k+1$.

Conclusion. The inequality holds for every $n \geq 0$. $\square$

Proposition 4. Suppose $a_1,a_2,\ldots,a_n$ is a sequence. For every distinct $x,y,z \in \mathbb{N}$, i.e. $x \neq y, y \neq z$ and $z \neq x$, if $next(x) < next(y)$, then $f_{z,y}(a) \geq f_{z,x}(a)$.

Proof: By induction on the length n of sequence a.

Basis. It is trivial for the case $n=0$ since the both hand sides are 0.

Induction. Suppose true for the case $n=k(k\geq 0)$, i.e.

$$f_{z,y}(a) \geq f_{z,x}(a).$$

holds for every sequence a of length k. Now consider a sequence $a_1,a_2,\ldots,a_{k+1}$ of length $k+1$.

Case 1. $a_1=z$. By Proposition 1 and 3, the left hand side becomes

$$\max\{f_{z,y}(a_2,\ldots,a_{k+1}), f_{z,z}(a_2,\ldots,a_{k+1})+1\} = f_{z,z}(a_2,\ldots,a_{k+1})+1,$$

and the right hand side becomes

$$\max\{f_{z,x}(a_2,\ldots,a_{k+1}), f_{z,z}(a_2,\ldots,a_{k+1})+1\} = f_{z,z}(a_2,\ldots,a_{k+1})+1$$

The inequality holds immediately.

Case 2. $a_1=x$. By Proposition 1, the left hand side becomes

$$\max\{f_{x,y}(a_2,\ldots,a_{k+1})+1, f_{z,x}(a_2,\ldots,a_{k+1})+1\},$$

and the right hand side becomes

$$\max\{f_{x,x}(a_2,\ldots,a_{k+1})+1, f_{z,x}(a_2,\ldots,a_{k+1})\}.$$

By Proposition 2, we have

$$f_{z,x}(a_2,\ldots,a_{k+1}) \geq f_{x,x}(a_2,\ldots,a_{k+1}),$$

and therefore, the inequality holds.

Case 3. $a_1=y$. This is impossible because $next(x) < next(y)$, i.e. there is an element of value x in front of the first element of value y.

Case 4. $a_1 \notin \{x,y,z\}$. The left hand side becomes

$$\max\{f_{a_1,y}(a_2,\ldots,a_{k+1})+1, f_{a_1,z}(a_2,\ldots,a_{k+1})+1\}.$$

The right hand side becomes

$$\max\{f_{a_1,x}(a_2,\dots,a_{k+1})+1, f_{a_1,z}(a_2,\dots,a_{k+1})+1\}.$$

Case 4.1. If $next(y) > next(z)$, then by induction we have

$$f_{a_1,y}(a_2,\dots,a_{k+1}) \geq f_{a_1,z}(a_2,\dots,a_{k+1}),$$

and (because $next(y) > next(x)$,)

$$f_{a_1,y}(a_2,\dots,a_{k+1}) \geq f_{a_1,x}(a_2,\dots,a_{k+1}).$$

The inequality holds.

Case 4.2. If $next(y) < next(z)$, then by induction we have

$$f_{a_1,z}(a_2,\dots,a_{k+1}) \geq f_{a_1,y}(a_2,\dots,a_{k+1}),$$

and (because $next(z) > next(y) > next(x)$,)

$$f_{a_1,z}(a_2,\dots,a_{k+1}) \geq f_{a_1,x}(a_2,\dots,a_{k+1}).$$

The inequality holds.

The inequality holds for all cases. Therefore, the inequality holds for $n = k+1$.

Conclusion. The inequality holds for every $n \geq 0$.

Proposition 5 (Greedy Strategy I). Suppose $a_1, a_2, \dots, a_n$ is a sequence. For every $x, y \in \mathbb{N}$, if $a_1 = x$, then

$$f_{x,y}(a_1,\dots,a_n) = f_{x,x}(a_2,\dots,a_n) + 1.$$

Proof: By Proposition 1, we have

$$f_{x,y}(a_1,\dots,a_n) = \max\{f_{x,y}(a_2,\dots,a_n), f_{x,x}(a_2,\dots,a_n)+1\}.$$

By Proposition 3, we have

$$f_{x,x}(a_2,\dots,a_n) + 1 \geq f_{x,y}(a_2,\dots,a_n).$$

Combining with the both above yields the proof. $\square$

Proposition 6 (Greedy Strategy II). Suppose $a_1, a_2, \dots, a_n$ is a sequence. For every $x, y \in \mathbb{N}$ with $x \neq y$, if $a_1 \notin \{x, y\}$, then

$$f_{x,y}(a_1,\dots,a_n) = \begin{cases} f_{a_1,y}(a_2,\dots,a_n)+1 & next(x) < next(y), \\ f_{x,a_1}(a_2,\dots,a_n)+1 & \text{otherwise.} \end{cases}$$

Proof: If $next(x) < next(y)$, by Proposition 4, we have

$$f_{a_1,y}(a_2,\dots,a_n) \geq f_{x,a_1}(a_2,\dots,a_n).$$

Therefore, by Proposition 1, we have $f_{x,y}(a_1,\dots,a_n) = f_{a_1,y}(a_2,\dots,a_n) + 1$.

The same statement holds for $next(x) > next(y)$. $\square$

1479B2 - Painting the Array II

Tutorial

[problem:1479B2]

There are two approaches from different perspectives.

DP Approach

The first observation is that merging adjacent elements with the same value will not influence the answer. Therefore, without loss of generality, we may assume that there are no adjacent elements with the same value, i.e. $a_i \neq a_{i+1}$ for every $1 \leq i < n$.

We can solve this problem by a DP approach. Let $f(i)$ denote the minimal possible number of segments for sub-array $a_1, a_2, \dots, a_i$ over all assignments $b_1, b_2, \dots, b_i$ with $b_i \neq b_{i-1}$, where $b_0 = -1$ for convenience. To obtain the answer, we enumerate the last position $1 \le i \le n$ such that $b_{i-1} \neq b_i$, and append all elements $a_{i+1}, a_{i+2}, \dots, a_n$ to the end of a_i, which implies an arrangement with $f(i) + n - i$ segments. The minimal number of segments will be the minimum among $f(i) + n - i$ over all $1 \le i \le n$.

It is straightforward to see that $f(0) = 0$ and $f(1) = 1$.

For $2 \le i \le n$, $f(i)$ can be computed by enumerating every possible position $1 \le j < i$ such that $b_{j-1} \neq b_j = b_{j+1} = \cdots = b_{i-1} \neq b_i$. That is, $a_j, a_{j+1}, \dots, a_{i-1}$ are assigned to the same sub-sequence, and a_{j-1} and a_i are assigned to the other sub-sequence. Since no adjacent elements has the same value (by our assumption), there are $(i - j)$ segments in $a_j, a_{j+1}, \dots, a_{i-1}$ (we note that the first segment, i.e. the segment of a_j, is counted in $f(j)$). Moreover, there will be zero or one new segment when concatenating a_{j-1} and a_i depending on whether $a_{j-1} = a_i$ or not. Hence, for every $2 \le j \le n$, we have

$$f(i) = \min_{1 \le j < i} \{f(j) + (i - j - 1) + [a_{j-1} \neq a_i]\},$$

where $[\text{Boolean expression}] = 1$ if the Boolean expression is true and 0 otherwise. Here, we obtain an $O(n^2)$ DP solution.

To optimize the DP recurrence, we fix i, and let $g(j) = f(j) + (i - j - 1) + [a_{j-1} \neq a_i]$, then $f(i) = \max_{1 \le j < i}\{g(j)\}$. We can observe that

Lemma 1. For $2 \le i \le n$, we have

$$f(i) = \min\{g(i-1), g(j^*)\},$$

where $j^* = \max\{1 \le j < i : a_{j-1} = a_i\}, \max \emptyset = 0$, and $g(0) = +\infty$.

This lemma is very intuitive, which means we need only to consider two cases: one is to just append a_i after a_{i-1} in the same sub-sequence, and the other is to append a_i after the closest a_j with the same value, i.e. $a_i = a_j$, and then assign the elements between them (not inclusive) to the other sub-sequence. With this observation, we immediately obtain an $O(n)$ DP solution.

The proof is appended below for completeness.

Proof: For every $1 \le j < i$, we have

$$f(i) \le g(j) = f(j) + (i - j - 1) + [a_{j-1} \neq a_i] \le f(j) + i - j,$$

which implies that $f(i) - i \le f(j) - j$ for every $1 \le j \le i \le n$.

Now we consider $g(j)$ for every $1 \le j < i$ in two cases.

1. $a_{j-1} \neq a_i$. We have

$$\begin{aligned} g(j) &= f(j) + (i - j - 1) + 1 \\ &= f(j) - j + i \\ &\ge f(i-1) - (i-1) + i \\ &= f(i-1) + 1 \\ &\ge g(i-1). \end{aligned}$$

2. $a_{j-1} = a_i$. Suppose there are two different positions j_1 and j_2 such that $1 \le j_1 < j_2 < i$ and $a_{j_1-1} = a_{j_2-2} = a_i$, then

$$\begin{aligned} g(j_1) &= f(j_1) + (i - j_1 - 1) \\ &= f(j_1) - j_1 + i - 1 \\ &\geq f(j_2) - j_2 + i - 1 \\ &= g(j_2). \end{aligned}$$

Combine the two cases, we conclude that $f(i) = \min\{g(i-1), g(j^*)\}$, where $j^* = \max\{1 \leq j < i : a_{j-1} = a_i\}$. $\square$

Greedy Approach

Consider we have a computer whose cache has only two registers. Let's suppose the array a is a sequence of memory access to the computer. The problem is then converted to a more intuitive one that asks the optimal cache replacement.

Suppose the current two registers contains two memory accesses x and y, and the current requirement memory access is z.

The greedy strategy is simple: If z matches one of x and y, just do nothing. Otherwise, the register that contains the memory access whose next occurrence is farther than the other will be replaced by z.

This strategy is know as Bélády's algorithm or farthest-in-the-future cache/page replacement policy (see here for more information). The complexity is $O(n)$ since we only need to preprocess every element's next occurrence.

1479C - Continuous City

Tutorial

1479D - Odd Mineral Resource

Tutorial

1479E - School Clubs

Tutorial

Tutorial of Codeforces Round #700 (Div. 1)

#tutorial

+7 liouzhou_101 72 minutes ago 33

Comments (33)

Write comment?

qinyihao

70 minutes ago, # | ← Rev. 2 -9

Thank you for high quality problems! Got suck on Div2 problem C and now found that it was only a binary search.

→ Reply

JackF

64 minutes ago, # | ← Rev. 2 -16

Will this round be unrated?

→ Reply

JackF

53 minutes ago, # ^ | 0

The problem A is same as find-local-minima-array. The problem B2 is similar to CF802A.

→ Reply

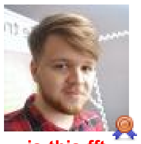
64 minutes ago, # | ← Rev. 2 -36

→ Reply

→ Reply

HimanshuG

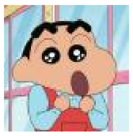
The comment is hidden because of too negative feedback, click here to view it


-is-this-fft-

57 minutes ago, # ^ | ☆ +15

Huh? The contest has exactly one constructive problem.

→ Reply


PR_0202

58 minutes ago, # | ☆ +1

Div 1A/2C was a nice use of binary search!

→ Reply

cpforfunCmnt

55 minutes ago, # ^ | ☆ +2

but was available on gfg

→ Reply

Eurus7

47 minutes ago, # ^ | ☆ 0

Yes but sadly very popular question easily googleable.

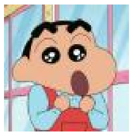
→ Reply

skittles1412

54 minutes ago, # | ☆ +26

Thank you for the original problems and strong pretests.

→ Reply


PR_0202

45 minutes ago, # ^ | ☆ +4

Now I understood the *hidden meaning* of this line.. xD

- antontrygubO_o for ~~rejecting problems~~ his amazing coordination and discussion.

→ Reply

shahi_panner

33 minutes ago, # ^ | ☆ 0

Nope, Pretests were useless!

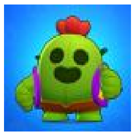
→ Reply

pksingh290

10 minutes ago, # ^ | ☆ 0

contest was good but the pretests were useless .

→ Reply


ankurkayal

49 minutes ago, # | ☆ 0

Thank you for the speedy and high-quality editorial with elaborate proofs :) . Keep up the good work!

→ Reply

ecnerwala

48 minutes ago, # | ☆ ← Rev. 2 0

For problem D, you don't need to consider an xor of 4 paths; we just have $f(u,v,l,r) = f(1,u,l,r) \oplus f(1,v,l,r) \oplus \mathbf{1}_{A_{lca(u,v)}}$; then you can special case $A_{lca(u,v)}$ and the rest just needs to find one unequal index, which doesn't require any special mod 2 properties of XOR.

→ Reply

→ Reply

Traduttore

48 minutes ago, # | ☆ ← Rev. 2 +3

Hello.

What is the answer for this test (Div.2 B) :

1 1 10

1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000

1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000

→ Reply

tankman890

47 minutes ago, # | ☆ 0

Can anyone tell me why my solution for Div2B is wrong? (https://codeforces.com/contest/1480/submission/106778511)

→ Reply

Snow

43 minutes ago, # | ☆ 0

The problem Div 1A/2C , could be many local minimals ?

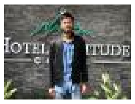
→ Reply

asfd221

43 minutes ago, # | ☆ 0

Div1 B caught me completely off guard.

→ Reply


dr.stone

33 minutes ago, # | ☆ 0

good problems but weak pretests

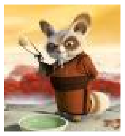
→ Reply

galen_colin

30 minutes ago, # | ☆ +4

If you like videos, here's one that happens to have all the div. 2 solutions

→ Reply


Olerinskiy

28 minutes ago, # | ☆ +6

Nice Div1 B1 editorial

→ Reply

dhruv7888

27 minutes ago, # | ☆ 0

Man my Div2 C/Div1 A Binary Search solution with 5*Log2(n) gave TLE on test 11 this is not right man. can anyone tell me the reason for TLE in this solution https://codeforces.com/contest/1480/submission/106790379

→ Reply

vineet16

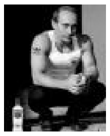
23 minutes ago, # | ☆ 0

Was anyone able to figure out why TC 11 of problem C gives FST?

→ Reply

19 minutes ago, # ^ | ☆ 0

I think mine failed on 1-based index.


EnterYourName

I think mine failed on 1-based index.
→ Reply

utr491

17 minutes ago, # ^ | ☆ 0

You are not storing the values after the query. This results in repeated queries. I did the same thing. :`(
→ Reply

vineet16

12 minutes ago, # ^ | ☆ 0

Yeah but I'm using only 3 queries at max. each time before reducing binary search range,which happens 18 times(worst case). That is still less than 60 queries.
→ Reply

utr491

9 minutes ago, # ^ | ☆ 0

That's exactly what I thought. But the accepted solutions I have seen differ from mine in just this one aspect. Well the sys tests are almost over. Let's see what was the reason.
→ Reply

treewave

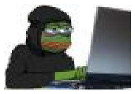
7 minutes ago, # ^ | ☆ 0

the test case is 1
→ Reply

WizardMan

15 minutes ago, # | ☆ 0

Thank you very much it will help us
→ Reply


abhishek945

14 minutes ago, # | ☆ +1

What's the proof of div2C why is binary search working here
→ Reply

utr491

a moment ago, # ^ | ☆ 0

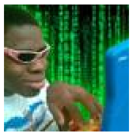
Imagine it as hills. If you are at a valley, you don't need to move. If you are at a peak, you can go in either direction. If you are at a slope, just go down the slope until you get to a valley.
→ Reply

SHZhang2

13 minutes ago, # | ☆ +2

Can **hos.lyric** and **maroonrk** explain their solutions to Div1E? It seems that they used different solutions from the one in the editorial (which uses very advanced generating function techniques).

(I know **maroonrk**'s solution got TLE, but the fact that it only failed on test 76 indicates that it is probably a correct solution with a high constant factor.)
→ Reply



2 minutes ago, # | ☆ ← Rev. 2 0

I haven't seen anyone else fail for test 55 for problem C, could someone point me in the right direction/test case as to why my solution is wrong? I can't see the test for it as of yet

sskhynix

submission:

I've considered the case for n == 1 & 2.

I'm not using an array to store the values which apparently could lead to repeated queries, but people seem to be getting a TLE in an earlier test case for this.

Perhaps my binary search implementation is skipping over an element for the minimum.

As always, apologies for the god awful formatting, it's my first time solving an interactive problem.

→ Reply