



Endagorion's blog

Codeforces Round #283: editorial (with bonuses!)

 By [Endagorion](#), 21 month(s) ago, translation, , 

Each problem comes with a challenge — a bonus task somehow related to the problem; you may tackle at the challenges for fun and practice, also feel free to discuss them at the comments. =)

496A - Minimum Difficulty

For every option of removing an element we run through the remaining elements and find the maximal difference between adjacent ones; print the smallest found answer. The solution has complexity $O(n^2)$. It can be noticed that after removing an element the difficulty either stays the same or becomes equal to the difference between the neighbours of the removed element (whatever is larger); thus, the difficulty for every option of removing an element can be found in $O(1)$, for the total complexity of $O(n)$. Any of these solutions (or even less efficient ones) could pass the tests.

Challenge: suppose we now have to remove exactly k arbitrary elements (but the first and the last elements have to stay in their places). How small the maximal difference between adjacent elements can become? Solve this problem assuming the limitations are as follows: $1 \leq k \leq n - 2$, $n \leq 10^5$, $a_i \leq 10^9$.

496B - Secret Combination

We observe that the order of operations is not important: we may first perform all the shifts, and after that all the additions. Note that after n shifts the sequence returns to its original state, therefore it is sufficient to consider only the options with less than n shifts. Also, after 10 times of adding 1 to all digits the sequence does not change; we may consider only options with less than 10 additions. Thus, there are overall $10n$ reasonable options for performing the operations; for every option perform the operations and find the smallest answer among all the options. As performing the operations for every option and comparing two answers to choose the best takes $O(n)$ operations, this solution performs about $10n^2$ elementary operations. The multiple of 10 can be get rid of, if we note that after all shifts are made the best choice is to make the first digit equal to zero, and this leaves us but a single option for the number of additions. However, implementing this optimization is not necessary to get accepted.

Challenge: can you solve the problem in $O(n \log n)$ time? in $O(n)$ time?

496C - Removing Columns/497A - Removing Columns

Let's look at the first column of the table. If its letters are not sorted alphabetically, then in any valid choice of removing some columns it has to be removed. However, if its letters are sorted, then for every valid choice that has this column removed it can be restored back to the table; it is clear that the new choice is valid (that is, the rows of the new table are sorted lexicographically) and the answer (that is, the number of removed columns) has just became smaller.

Consider all columns from left to right. We have already chosen which columns to remove among all the columns to the left of the current one; if leaving the current column in place breaks the lexicographical order of rows, then we have to remove it; otherwise, we may leave it in place to no harm. Arguing in the way of the previous paragraph we can prove that this greedy method yields an optimal (moreover, the only optimal) solution. The complexity is $O(n^2)$.

→ Pay attention

Before contest

[Codeforces Round #370 \(Div. 2\)](#)
10 days

→ WuHongxun

Rating: **2398**
Contribution: **0**



WuHongxun

- [Settings](#)
- [Blog](#)
- [Favourites](#)
- [Teams](#)
- [Submissions](#)
- [Talks](#)
- [Contests](#)

→ Top rated

| # | User | Rating |
|----|---------------------------------|--------|
| 1 | tourist | 3580 |
| 2 | Petr | 3481 |
| 3 | TooDifficult | 3317 |
| 4 | rng_58 | 3102 |
| 5 | ainta | 3048 |
| 6 | dotorya | 3046 |
| 7 | PavelKunyavskiy | 3007 |
| 8 | jcvb | 2999 |
| 9 | vepifanov | 2992 |
| 10 | RomaWhite | 2982 |

[Countries](#) | [Cities](#) | [Organizations](#) | [View all →](#)

→ Top contributors

| # | User | Contrib. |
|---|---------------------------|----------|
| 1 | gKseni | 176 |
| 2 | Errichto | 175 |
| 3 | Petr | 166 |
| 4 | Zlobober | 162 |
| 5 | Swistakk | 157 |
| 5 | Edvard | 157 |
| 7 | rng_58 | 154 |
| 8 | csacademy | 147 |
| 9 | Xellos | 146 |
| 9 | chrome | 146 |

[View all →](#)

→ Find user

Handle:

Find

Challenge: compute how many (say, modulo $10^9 + 7$) $n \times m$ tables are there for which the answer for this problem is k ? The more efficient solution you come up with, the better.

496D - Tennis Game/497B - Tennis Game

Choose some t ; now emulate how the match will go, ensure that the record is valid for this t and by the way find the corresponding value of s . Print all valid options for s and t . This solution works in $O(n^2)$ time, which is not good enough, but we will try to optimize it.

Suppose the current set is finished and we have processed k serves by now. Let us process the next set as follows: find t -th 1 and t -th 2 after position k . If t -th 1 occurs earlier, then the first player wins the set, and the set concludes right after the t -th 1; the other case is handled symmetrically. If the match is not over yet, and in the rest of the record there are no t ones nor t twos, then the record is clearly invalid. This way, every single set in the record can be processed in $O(\log n)$ time using binary search, or $O(1)$ time using precomputed arrays of positions for each player.

Now observe that for any t a match of n serves can not contain more than n/t sets, as each set contains at least t serves. If we sum up the upper limits for the number of sets for each t , we obtain the total upper limit for the number of sets we may need to process: $n + n/2 + n/3 + \dots = O(n \log n)$ (which is the famous harmonic sum). Using one of the approaches discussed above, one obtains a solution with complexity of $O(n \log^2 n)$ or $O(n \log n)$; each of these solutions fits the limit nicely.

Obviously, for every t there is no more than one valid choice for s ; however, maybe a bit unexpected, for a given s there may exist more than one valid choice of t . The first test where this takes place is pretest 12. The statement requires that the pairs are printed lexicographically ordered; it is possible to make a mistake here and print the pairs with equal s by descending t (if we fill the array by increasing t and then simply reverse the array).

Challenge: while preparing this problem I discovered that it's quite hard to find a test such that the number of pairs in the answer is large; in the actual tests the maximal number is 128, which is the number of divisors of the number 83160. Can you beat this record? If you have a test with $n \leq 10^5$ that has larger number of pairs in the answer, feel free to brag in the comments; also don't hesitate to share any insights on how one could bound the maximal number analytically.

496E - Distributing Parts /497C - Distributing Parts

Sort all the parts and actors altogether by increasing lower bounds (if equal, actors precede parts); process all the entities in this order. We maintain a set of actors which have already occurred in the order; if we meet an entry for an actor, add it to the set. If we currently process a part, we have to assign it to an actor; from the current set of actors we have to choose one such that his $d_i \geq b_j$ (the $c_i \leq a_j$ constraint is provided by the fact that the i -th actor has occurred earlier than the j -th part); if there are no such actors in the set, no answer can be obtained; if there are several actors satisfying this requirement, we should choose one with minimal d_i (intuitively, he will be less useful in the future). Assign the chosen actor with the current part and decrement his k_i ; if k_i is now zero, the actor can not be used anymore, thus we remove him from the set.

To fit the limits we should implement the set of current actors as some efficient data structure (e.g., an `std::set` or a treap). The resulting complexity is $O((n+m) \log(n+m))$.

Challenge: suppose that now there are q_j copies of the j -th part ($1 \leq q_j \leq 10^9$), and each copy must be separately assigned with an actor in a valid way. Can you solve this new problem with all the old constraints (as the actual distribution now has too much entries, it is sufficient to check whether an answer exists)?

497D - Gears

When a collision happens, a vertex of one polygon lands on a side of the other polygon. Consider a reference system such that the polygon A is not moving. In this system the polygon B preserves its orientation (that is, does not rotate), and each of its vertices moves on some circle. Intersect all the circles for vertices of B with all the sides of A ; if any of them intersect, then some vertex of B collides with a side of A . Symmetrically, take a reference system associated with B and check whether some vertex of A collides with a side of B .

→ Recent actions

MikeMirzayanov → [Codeforces: C++14 is supported](#) 🔔

zloyplace35 → [Codeforces Round #368 \(Div.2\) па360p](#) 🔔

cjtoribio → [Unknown Data Structure — \(Sqrt Fragmented Tree\)](#) 🔔

adamant → [Palindromic tree: behind the scenes](#) 🔔

Edvard → [Editorial of Educational Codeforces Round 16](#) 🔔

cgy4ever → [Codeforces Round #190 — Editorial](#) 🔔

BYN → [Bayan Contest Warm Up round + 50 tshirts!](#) 🔔

zscoder → [Codeforces Round #369 Editorial](#) 🔔

zscoder → [Codeforces Round #369 \(Div. 2\)](#) 🔔

Nickolas → [TCO16 Russia regional event at ITMO](#) 🔔

appa → [My name is Hongjun Jang.](#) 🔔

ifsmirnov → [On Euler tour trees](#) 🔔

Edvard → [Editorial of Educational Codeforces Round 2](#) 🔔

Iewin → [Invitation to HackerEarth August Circuits](#) 🔔

memset123 → [Invitation to HackerEarth September Easy](#) 🔔

SkidanovAlex → [MemSQL Start\[c\]UP 2.0 Round 1 and 2 Editorials](#) 🔔

Arpa → [\[Tutorial\] dsu on tree](#) 🔔

Edvard → [Разбор задач Educational Codeforces Round 9](#) 🔔

Antoniuk → [Editorial Codeforces Round #266 \(Div. 2\)](#) 🔔

reality → [Codeforces Round #361 \(Div. 2\) Editorial](#) 🔔

Edvard → [Educational Codeforces Round 16](#) 🔔

ahmed_aly → [A2OJ BOT](#) 🔔

Edvard → [Editorial of Educational Codeforces Round 14](#) 🔔

Kostroma → [Codeforces Aim Tech Round 3 Editorial](#) 🔔

Shafaet → [HackerRank World Codesprint May](#) 🔔

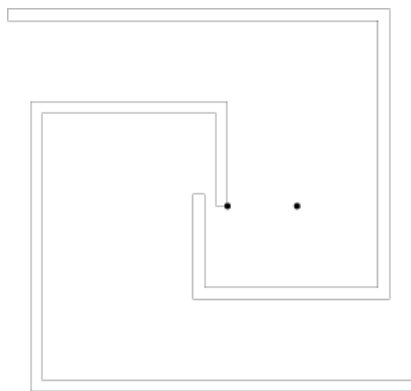
[Detailed →](#)

The constraints for the points' coordinates are small enough for a solution with absolute precision to be possible (using built-in integer types).

Another approach (which is, in fact, basically the same) is such: suppose there is a collision in a reference system associated with A . Then the following equality for vectors holds: $x + y = z$; here z is a vector that starts at P and ends somewhere on the bound of A , x is a vector that starts at Q and ends somewhere on the bound of B , y is a vector that starts at P and ends somewhere on the circle centered at P that passes through Q . Rewrite the equality as $y = z - x$; now observe that the set of all possible values of $z - x$ forms the Minkowski sum of A and reflection of B (up to some shift), and the set of all possible values of y is a circle with known parameters. The Minkowski sum can be represented as a union of nm parallelograms, each of which is the Minkowski sum of a pair of sides of different polygons; finally, intersect all parallelograms with the circle.

Both solutions have complexity $O(nm)$. As noted above, it is possible to solve the problem using integer arithmetics (that is, with absolute precision); however, the fact that the points' coordinates are small lets most of the solutions with floating point arithmetics pass. It was tempting to write an approximate numerical solution; we struggled hard not to let such solutions pass, and eventually none of them did. =)

Many participants had troubles with pretest 8. It looks as follows (the left spiral revolves around the left point, and the right spiral revolves around the right point):



Challenge: suppose we want a solution that uses floating point arithmetics to fail. In order to do that, we want to construct a test such that the polygons don't collide but pass really close to each other. How small a (positive) distance we can achieve, given the same constraints for the number of points and the points' coordinates?

497E - Subsequences Return

Consider some string; how does one count the number of its distinct subsequences? Let us append symbols to the string consequently and each time count the number of subsequences that were not present before. Let's append a symbol c to a string s ; in the string $s + c$ there are as many subsequences that end in c as there were subsequences in s overall. Add all these subsequences to the number of subsequences of s ; now each subsequence is counted once, except for the subsequences that end in c but were already present in s before; these are counted twice. Thus, the total number of subsequences in the new string is twice the total number of subsequences in the old string minus the number of subsequences in the old string which end in c .

This leads us to the following solution: for each symbol c store how many subsequences end in c , denote cnt_c . Append symbol c ; now cnt_c becomes equal to the sum of all cnt 's plus one (for the empty subsequence), and all the other cnt 's do not change.

For example, consider the first few symbols of the Thue-Morse sequence:

- $\varepsilon \rightarrow (0, 0)$
- $0 \rightarrow (0 + 0 + 1 = 1, 0)$
- $01 \rightarrow (1, 1 + 0 + 1 = 2)$
- $011 \rightarrow (1, 1 + 2 + 1 = 4)$
- $0110 \rightarrow (1 + 4 + 1 = 6, 4)$

• ...

Let us put the values of cnt in the coordinates of a vector, and also append a coordinate which is always equal to 1. It is now clear that appending a symbol to the string alters the vector as a multiplication by some matrix. Let us assign a matrix for each symbol, and also for each string as a product of matrices for the symbols of the strings in that order.

Now, consider the prefix of the sequence a_i of length k^m . Divide it into k parts of length k^{m-1} ; x -th (0-based) of these parts can be obtained from the 0-th one by adding x modulo k to all elements of the part. Let us count the matrices (see above) for the prefixes of length k^m , and also for all strings that are obtained by adding x to all of the prefixes' elements; denote such matrix $A_{m,x}$.

It is easy to see that if $m > 0$, then

$A_{m,x} = A_{m-1,x} A_{m-1,(x+1) \bmod k} \dots A_{m-1,(x+k-1) \bmod k}$. This formula allows us to count $A_{m,x}$ for all $m \leq \log_k n$ and all x from 0 to $k-1$ in $O(\log_k n \times k \times k \times k^3) = O(\log n \times k^5 / \log k)$ time. Now, upon having all $A_{m,x}$ we can multiply some of them in the right order to obtain the matrix for the prefix of the sequence a_i of length n .

Unfortunately, this is not quite enough as the solution doesn't fit the time limit yet. Here is one way to speed up sufficiently: note that the product in the formula

$A_{m,x} = A_{m-1,x} A_{m-1,(x+1) \bmod k} \dots A_{m-1,(x+k-1) \bmod k}$ can be divided as shown:

$A_{m-1,x} \dots A_{m-1,k-1} \times A_{m-1,0} \dots A_{m-1,x-1}$ (if $x = 0$, take the second part to be empty).

Count all the "prefixes" and "suffixes" products of the set $A_{m,x}$: $P_{m,x} = A_{m,0} \dots A_{m,x-1}$, $S_{m,x} = A_{m,x} \dots A_{m,k-1}$. Now $A_{m,x} = S_{m-1,x} P_{m-1,x}$. Thus, the computation of $A_{m,x}$ for all x and a given m can be done as computing all $P_{m-1,x}$, $S_{m-1,x}$ using $O(k)$ matrix multiplications, and each $A_{m,x}$ is now can be found using one matrix multiplication. Finally, the solution now works in $O(\log n \times k^4 / \log k)$ time, which fits the limits by a margin.

Challenge: solve the problem for $k \leq 100$.

 Tutorial of Codeforces Round #283 (Div. 2)

 +336 



[Endagorion](#)



21 month(s) ago



[36](#)



Comments (36)

[Write comment?](#)



[I_Love_Monkey](#)

20 months ago, <#> | ☆

 +1 

Can you describe how to solve 496B in $O(N)$ or $O(n \log n)$?

→ [Reply](#)



[dragonslayerx](#)

20 months ago, <#> [^](#) | ☆

 0 

Finding the lexicographically smallest rotation of a string using suffix array is $O(n)$.. and one can construct a suffix array in $O(n)$ so it can be done in $10n$ i.e $O(n)$

→ [Reply](#)



[thebvog](#)

20 months ago, <#> [^](#) | ☆

 0 

Lyndon decomposition can find smallest rotation of a string in $O(n)$ too, and it's more easy to coding.

→ [Reply](#)



[fluxay](#)

20 months ago, <#> [^](#) | ☆

 0 

Can you give some details? How to construct a suffix array in $O(n)$? Isn't it $O(n \log n)$? And how to find the smallest rotation in $O(n)$?

→ [Reply](#)



I_Love_Monkey

20 months ago, # ^ | ☆

▲ 0 ▼

Same before, I want you to describe more clearly about algorithm to solve in complexity $O(n)$ or $O(n \log n)$. I don't know how to construct a suffix array :/

→ Reply



adamant

20 months ago, # ^ | ☆

▲ 0 ▼

For example we can construct suffix automaton on string $S + S$ and then find the smallest substring of length n .

→ Reply



I_Love_Monkey

20 months ago, # ^ | ☆

▲ 0 ▼

And the complexity is about: $O(2^n * 10)$?

→ Reply



Mohamed_Alaa_Kandeel

20 months ago, # | ☆

▲ 0 ▼

for problem 496B Secret Combination i got wrong answer on test 11 while it gives me the expected answer on my PC here's the submission 9181590

→ Reply



Novik_D

20 months ago, # ^ | ☆

▲ 0 ▼

This is the end of your answer: 00201196. This is the end of right answer: 20201196.

→ Reply



Mohamed_Alaa_Kandeel

20 months ago, # ^ | ☆

▲ +1 ▼

thx i actually solved it. the problem was i thought that the output is the right answer and the answer is the code output :D

→ Reply



LouisCK

20 months ago, # | ☆

← Rev. 2

▲ 0 ▼

Can someone explain Div 2 C in english?

→ Reply



Kira96

20 months ago, # ^ | ☆

▲ +13 ▼

Basically, you have to iterate through columns from 1 to m , and compare two consecutive elements on the same column. Now, if we have $a[\text{row}][\text{col}] < a[\text{row}-1][\text{col}]$, then if we don't have a previous column k in the range $[1, \text{col}-1]$ where $a[\text{row}][k] > a[\text{row}-1][k]$, we must delete the column col and proceed to the next column.

→ Reply



LouisCK

20 months ago, # ^ | ☆

▲ 0 ▼

Thanks, what is an efficient way to find that column k though?

→ Reply

20 months ago, # ^ | ← Rev. 3

▲ 0 ▼



Kira96

First, you can notice that we only need the first k to satisfy this, so if $a[\text{row}][k] > a[\text{row}-1][k]$ and if we don't delete this column, then we take an array $OK[\text{row}] = 1$, and from now on if $a[\text{row}][\text{col}] < a[\text{row}-1][\text{col}]$ it won't matter as long as $OK[\text{row}]$ is already 1.

→ Reply

20 months ago, # ^ | ☆

▲ +6 ▼

You may translate this in Russian in case you wish that

you may translate this in Russian in case you wish that.

P.s. This is my 1st ever so ignore mistakes(not algorithmic ofc)

The problem:: We are given a table of letters and we need to ensure that any row i is NOT lexicographically less than the previous row $i-1$.

Operation:: We are allowed to delete any column we wish to.

Output

An integer which represents **minimum** number of columns that will have to be deleted to solve the problem.

STRATEGY=>GREEDY ::

We iterate from 0th column to last , deciding for each column whether it will be removed or included in our final table.

2 variables- prevtaken,prevleft which will store the index of last column which was included,removed respectively.**Both -1 initially.**

answer=number of 0s columns included :P obvio



dreamplay

Say a function $f(i)$ = **true** if while moving **down** the i th column the letters are in lexicographic order and **FALSE** otherwise

Algorithm::start iterating from left

All those columns for which $f(i)$ =>false&&prevtaken=>-1 are to be removed.(why? easy:p)

All those columns for which $f(i)$ =>>true are to be included(obvio? easy:P)

Now consider those columns for which $f(i)$ =>>false but prevtaken is not -1

As we iterate down this column finding the lexicographic order not satisfied, if for these 2 rows,if the elements in prevtaken column for these 2 rows are same then we remove this column(why? bcz otherwise the rowwise lexico order that we are interested in would have already been satisfied as prevtaken column is more significant).

Hopeefully you read the hopefully.. xD

P.s. The length is compromised so as to make it comprehensible for even a newbie.:)

<http://codeforces.com/contest/496/submission/9179405>

→ [Reply](#)



achut1993

20 months ago, # | ☆

▲ 0 ▼

can some one explain 496E distribution of parts in english ?

→ [Reply](#)



rvkmr102

20 months ago, # | ☆

▲ 0 ▼

Google Translator is quite good.

→ [Reply](#)



slowlight

20 months ago, # | ☆

▲ 0 ▼

How to solve the challenge of 496A?

→ [Reply](#)



ffao

20 months ago, # ^ | ☆

▲ +4 ▼

Binary search on final difficulty d , then remove as many holds as you can greedily without letting difficulty exceed d .

Answer is smallest d such that you can remove at least k holds.

→ Reply



slowlight

20 months ago, # ^ | ☆

▲ 0 ▼

Um...To check if d can be achieved, can we do this? let i point to the 1st element and j to the 2nd, remove $a[j]$ and inc j by 1 until $a[j+1]-a[i] > d$, then $i=j$, $j=j+1$...

→ Reply



ffao

20 months ago, # ^ | ☆

▲ 0 ▼

That's it. Can you prove that this always achieves d with the smallest amount of holds left?

→ Reply



slowlight

20 months ago, ← Rev. 2 ☆ ▲ 0 ▼

When i, j point to the 1st and 2nd hold respectively, we know that replacing j with a hold on its left wont make things better. For if you remove j after that, it is harder to remove next holder because the distance gets larger. And if you dont remove j , the situation becomes the same before replacing. Thank u, ffao, my mind is clear with this greed approach now :)

→ Reply



R.A.X

18 months ago, # ^ | ☆

▲ 0 ▼

How to solve challenge problem div 2 c??

→ Reply



saurv4u

20 months ago, # | ☆

▲ 0 ▼

what are precomputed arrays of positions of players in 496D/497B Tennis Game Div. 1 B which reduce the complexity to $O(n \log n)$?

→ Reply



Guliash

20 months ago, # ^ | ☆

← Rev. 2

▲ 0 ▼

score[i] — the score of a player for the first i serves.

step[i] — how much serves are past until a player reaches the score = i .

→ Reply

20 months ago, # ^ | ☆

▲ 0 ▼

$s1[i]$ = sum of points the player 1 has the index i of the main vector $A1[j]$ = index in the main vector that player 1 gets j points

There are the same vectors for player 2.

Using only the $s1$ and binary search is possible complexity $n * \log^2(n)$. Using $s1$ and $a1$ is possible complexity $n * \log(n)$.

→ Reply



jlr.terceiro

20 months ago, # | ☆

▲ 0 ▼

Can anyone explain in 496D - Tennis Game why is it wrong to say that number of valid (s, t) will be 0 if the total number of

points of player who won the last point is less than or equal

points of player who won the last point is less than or equal to another player



xpertcoder

Because if the game is valid(i.e we have some valid s and t pair) this means the player who won it must have won **strictly more** sets than other and hence more points.

WA -->9231337

AC after I removed that condition --> 9231351

→ [Reply](#)

20 months ago, # |

← Rev. 2

+6

Not necessarily. Let, $s = 2$, $t = 5$ and consider the following score:

A B

0 5

5 4

5 4

In this case, B's total point is 13 while A's is 10, but the winner is A :)

→ [Reply](#)



Hedayet

20 months ago, # |

0

Thnx, never thought of that :)

→ [Reply](#)



xpertcoder

20 months ago, # |

0

Div1D, why if we take A as the reference system, then the vertices on B are moving on a circle? I think the vertices on B are moving on a moving circle whose center is moving on a fixed circle. A is like sun, and B is moon, and our earth is reference system? Where am I wrong?497D -

Шестерни

→ [Reply](#)



ccdd4211

20 months ago, # |

0

Problem D div 1:

"In this system the polygon B preserves its orientation (that is, does not rotate), and each of its vertices moves on some circle"

What circle? Can anyone give more detailed explanation? Or what should I study to understand this?

→ [Reply](#)



I_love_Hoang_Yen

20 months ago, # |

← Rev. 3

+8

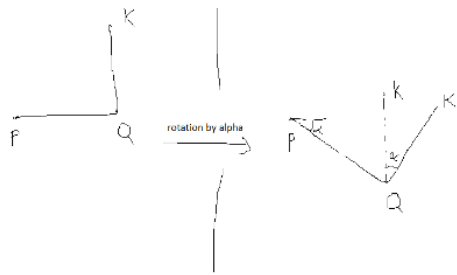
Let K be a point in polygon B. In the frame of reference in which A does not rotate, Q rotates around P in a circle (this should be easy to see).

What is a bit difficult to see is that, as the frame of reference is rotating counterclockwise around P and K is rotating clockwise around Q, K doesn't move relative to Q at all. Maybe this is easier to see with a picture:

ETA: I now noticed I accidentally switched the rotations so that Q is rotating clockwise and K is rotating counterclockwise in the picture. The principle still holds, though.



ffao



K' is where K would be after the rotation of the frame of reference. K is the final position after accounting for the fact that K is also rotating around Q with the same angular speed. As you can see, K is exactly at the same position relative to Q (directly above) in both pictures.

So K follows the exact same trajectory as Q , but with a constant translation.

→ [Reply](#)

20 months ago, # |

0

Thank you. Your explanation is very clear.



I_love_Hoang_Yen

Though only when 2 polygons rotate at the same angular speed, it has this nice property. What if 2 polygons rotate at different speed? How can it be solved?

→ [Reply](#)



.redo.

20 months ago, # |

0

I cannot understand Div 2 C correctly, please can you tell me what am I missing in the problem statement? Because I think I have to remove every column in which is the wrong order, but it is not the correct answer. Here is my code <http://codeforces.com/contest/496/submission/9290756>

→ [Reply](#)

20 months ago, # |

0

Hi everyone!

I had a question on Div 2 C. I've been looking through a lot of other peoples' solutions and it seems many people use a boolean array to store something about the previous "state" of the lexicographic ordering, and I'm not sure I understand the optimization.

My practice solution (after reading the editorial) built up intermediate strings based on which columns I decided to include, and used those to handle out of order characters in columns.

For example: ab f ba a

abf < baa, but I don't know how you could compress that into a single bool for each column.

For reference, my solution is #9340197 here:

http://codeforces.com/submissions/vk_.

A solution I was wondering about is #9164431 here:

<http://codeforces.com/submissions/chenmark>

→ [Reply](#)

16 months ago, # |

0



SabakuNoGaara

I have the same approach for problem "496D — Tennis Game" and I still have TLE for test 23. Can anyone tell me why? here is my code 11077258 I have explained some steps in comments there.

→ [Reply](#)

[Codeforces](#) (c) Copyright 2010-2016 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Aug/31/2016 20:08:54^{UTC+8} (c2).
Desktop version, switch to [mobile version](#).