Nerevar's blog

Codeforces Round #207: tutorial

By Nerevar, 3 years ago, translation,

357A - Group of Students

In this problem you need to iterate over all possible values of passing rate from 1 to 100 and for each value calculate the sizes of two groups.

357B - Flag Day

Let's process the dances in the given order and determine the colors of dancers' clothes. If there are no dancer from some previous dance, we can give the dances different colors arbitrarily. And if there is such dancer, we already know the color of his clothes. So, we arbitrarily distribute the other two colors between the remaning two dancers.

356A - Knight Tournament

Let's the current fight (l, r, x) consists of K knights fighting. Then all we have to do is to find all these knights in time $O(K)$ or $O(KlogN)$. There are several ways to do that, let's consider some of them.

The first way is to store the numbers of all alive knights in std::set (C++) or TreeSet (Java). Then in C++ we can use lower_bound method to find the first knight in the fight that is alive, and to iterate over this set, each time moving to the next alive knight. In Java we should use subSet method.

```
set<int> alive;
for (int i = 0; i < n; i++)
    alive.insert(i);

for (int i = 0; i < m; i++) {
    int l, r, x;
    scanf("%d%d%d", &l, &r, &x);
    l--, r--, x--;
    set<int>::iterator it = alive.lower_bound(l);
    vector<int> toErase;
    while(it != alive.end()){
        int cur = *it;
        if(cur > r)
            break;
        if(cur != x){
            toErase.pb(cur); answer[cur] = x;
        }
        it++;
    }

    for (size_t j = 0; j < toErase.size(); j++)
        alive.erase(toErase[j]);
}
```

The second way is to define array next with the following meaning:

- if knight v is alive, then $next[v] = v$;

→ Pay attention

Before contest
Codeforces Round #370 (Div. 2)
10 days

→ WuHongxun

Rating: 2398
Contribution: 0

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

WuHongxun

→ 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:

- if knight v is out of tournament, next[v] points to some knight u ($next[v] = u$), such that there are no alive knights between v and u;

To find the first alive knight starting from the knight v we need to follow this links until we find the first knight w with $next[w] = w$. In order not to pass the same links too many times, we will use the trick known as path compression (it is used in Disjoint Set Union). Note that you should handle the case when the current knight is the last knight and is out of tournament.

```
    int getNext(int v){
        if(next[v] == v)
            return v;
        return next[v] = getNext(next[v]);
    }

    ...

     int cur = getNext(l);
     while(cur <= r){
        if(cur == x){
            cur = cur + 1;
        }else{
            answer[cur] = x;
            next[cur] = cur + 1;
        }

        cur = getNext(cur);
    }
```

356B - Xenia and Hamming

Let's denote the length of the first string as $lenX$, the length of the second string as $lenY$. Let $L = LCM(lenX, lenY)$. It's obvious that L is a period of the long strings a and b, so we can find the distance of its' prefixes of length L and multiply the answer by $\frac{len(a)}{L}$. Let's fix the position i in the string x and think about all characters from the second string it will be compared with. It easy to conclude that it will be compared with such y_j that $i \equiv j \ (mod\ g)$, where $g = GCD(lenX, lenY)$. For each possible remainder of division by g and for each character c we can calculate $count(r, c)$ — the number of characters c that appear in y in such positions j that $j\ mod\ g = r$. When calculating the Hamming distance, the character x_i will be compared with exactly $count(i\ mod\ g, x_i)$ characters from y that are equal to it, all other comparisons will add one to the distance.

```
private void solve() {
        Scanner in = new Scanner(System.in);
        long n = in.nextLong(), m = in.nextLong();
        String x = in.next(), y = in.next();
        int lenX = x.length(), lenY = y.length();
        int g = gcd(lenX, lenY);
        long L = lenX * (long)lenY / g;
        long answer = L;
        int[][] count = new int[g][26];
        for (int j = 0; j < lenY; j++) {
            count[j % g][y.charAt(j) - 'a']++;
        }
        for (int i = 0; i < lenX; i++) {
            answer -= count[i % g][x.charAt(i) - 'a'];
        }
        System.out.println(answer * (n * lenX / L));
    }
```

356C - Compartments

In the problem you should come up with some right greedy algorithm. One of correct approaches acts as follows:

Find

→ Recent actions

MikeMirzayanov → Codeforces: C++14 is supported

zloyplace35 → Codeforces Round #368 (Div.2) разбор

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

lewin → 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

Detailed →

1. Firstly, it joins all "twos" and "ones" (to get "threes"). Several "ones" should be moved.
2. Then you should consider two cases depend on amounts of "ones" and "twos". If initially you have more "ones", you should try to join remaining after the first point "ones" into groups of three. If initially you have more "twos", you should try to join remaining after the first point "twos" into groups of three. You can get two "threes" from three "twos".
3. After the first point and the second point some "ones" or "twos" can remain. You shouldn't come up with common solution. Else you should just to consider all possible cases.

To solve the problem you should follow your common sense (is it greedy?). Writing naive solution (bfs search) for stress purposes is not so bad for proving correctness of your solution.

356D - Bags and Coins

It's easy to see that bags and their relations "lies directly in" should form directed forest. Each vertex should be given value c_i — the number of coins in the corresponding bag. Let's denote the sum of values c_j in the subtree of vertex i as f_i. The following conditions should be met:

- $f_i = a_i$
- then sum of f_i of roots equals s.

It's clear that one of the bags with largest a_i must be the root of some tree. It's quite easy to see that the solution exists if and only if there exists a subset $a_{i1}, a_{i2}, ..., a_{ik}$ such that $a_{i1} + a_{i2} + ... + a_{ik} = s$ and this subset contains at least one bag with the largest a_i. It's obvious that it is necessary condition, the sufficiency is also easy to see: let's suppose we have such subset. Then all bags from the subset, except one of the largest, will be roots of the signle-vertex trees (i.e. $c_i = a_i$ for them). All bags that are not in the subset we will consequentially put into the largest bag, forming the "russian doll" (this tree will be directed chain).

So, we reduced the task to the well-known subset-sum problem: from the items $a_1, a_2, ... a_n$ find the subset with the given sum s. This problem is NP-Complete, and with these constraints is solved in a following way: let $T(i, j) = 1$ if it is possible to obtain sum j using some of the first i items, and $T(i, j) = 0$ otherwise. Then $T(i,j) = T(i-1,j) \vee T(i-1, j-a_i)$. The i-th row of this table depends only on the previous row, so we don't have to store the whole table in memory. Also we should use the fact that the values of the table are zeroes and ones, and we can use bit compression and store each row in an array of int's of size $\lceil \frac{s}{32} \rceil$. To get the i-th row, we should calculate the bitwise OR of the previous row and the previous row shifted to the left by a_i positions. That is, we can find out whether it possible to obtain the sum s in approximately $\frac{ns}{32}$ operations. To find the actual way to obtain s, we need to use the following trick: for every possible sum j we will remember the value $first(j)$ — the number of such item that after considering this item it became possible to obtain j. This allows us to restore the solution.

356E - Xenia and String Problem

During the contest most of participants write the solutions that are very similar to the author's one. One of the author's solution uses hashes (but there exist solution without it), you can see short description of the solution below:

1. For each position i calculate with hashes the maximal value of L_i, such that substring $s[(i - L_i + 1)..(i + L_i - 1)]$ is Gray string. Also, calculate the maximal value P_i, that substring $s[(i - P_i + 1)..(i + P_i - 1)]$ differs from some Gray string in at most one position. You can see that $P_i \geq L_i$. If $P_i > L_i$, also remember position and letter in the position, that differs Gray string and the substring.

2. You can see, that if we don't need to change letters, then the answer for the problem is $\sum_{i=1}^{|s|} f(L_i)$, where $f(L) = 1^2 + 3^2 + 7^2 + ... + L^2$. So, calculate an answer without changes.

3. Next, iterate through all positions and letters in it. What is the new answer for the problem? Look at all Gray strings that occurs in our string and touches our fixed position. After we change this position the string will not be Gray string anymore (so we should subtract the squired length of the string from our answer). Look at all Gray

strings that differs in exactly fixed position from some substring of the string. If we change the letter in the position to the fixed letter, all such strings will be added to the answer (and we should add their squired lengths).

4. Summary, with P_i and L_i we need to calculate for each position and letter, how the answer differs if we change the letter in the position to the fixed one. For that reason we should use offline update (+=) on the segment. After the values will be calculated we can update our answer with all possible values.

Tutorial of Codeforces Round #207 (Div. 2)

tutorial, round, 207

+60 Nerevar 3 years ago 26

Comments (26)

Write comment?

3 years ago, # | 0

Hi, would you please help me to clarify the problem 356B — Xenia and Hamming " It it easy to conclude that it will be compared with such yj that i ≡ j (mod g), where g = GCD(lenX, lenY)."

How to get that conclude?

Thanks

→ Reply

BobYu

3 years ago, # ^ | +6

let's say ith character in the kth repetition of x is matched with jth character in the pth repetition of y

(k * x + i) mod gcd = i mod gcd (p * y + j) mod gcd = j mod gcd

→ Reply

nishant4

3 years ago, # ^ | 0

Got it. Thank you very much!

→ Reply

BobYu

14 months ago, # ^ | 0

I still can't get it. How do we conclude the gcd part? Would someone explain it in more details, please?

→ Reply

MAS1

12 months ago, # ^ | 0

Let us consider the ith character in the xth repetition of the first string. And let us assume that it is matched with the jth character of the yth repetition of the second string. Then - (x*Len1+i) = (y*Len2+j) Should be true. Where Len1 and Len2 are the lengths of the first and second strings respectively.

If g = gcd(Len1,Len2), then we can write the equation as - (x*Len1+i) %g = (y*Len2+j) %g Hence, we get- i%g = j%g

→ Reply

demon_cross

3 years ago, # | ← Rev. 2 +23

For DivI A, I think it is more convenient to use the `erase` method in `set` :

```
set<int>::iterator It = S.lower_bound(l[i]);
```

gen

```
set<int>::iterator lt = S.lower_bound(l[i]);
set<int>::iterator rt = S.upper_bound(r[i]);
for(set<int>::iterator it = lt; it != rt; it++) {
    int val = *it;
    if(val != x[i]) {
        ans[val] = x[i];
    }
}
S.erase(lt, rt);
S.insert(x[i]);
```
→ Reply

3 years ago, # | ☆ ← Rev. 3 +15

eduardische

I had a different solution for Div 1 C, which I think is easier to prove to yourself and harder to miss a case in.

Let's sort the numbers. Then, intuitively, it seems that there should exist a border in sorted train which splits the carriages into two parts — carriages with no students in it [first part], and carriages with 3 or 4 students in it [second part] (I would be very glad if someone could give me general proof for that [if more people were allowed in carriages and not just four] — maybe this isn't true in general case).

So let's check all possible border locations. Now that we have it fixed, let's denote four values, which can be computed in O(1) using partial sums:

- source — how many students are in first part (we need to move them),
- target — how many students are missing in the second part, such that every carriage in the second part has at least 3 students,
- free — how many free space there is in the second part,
- additional — how many students can be moved from the second part (aka how many carriages are there with 4 students)

Now, source <= free should hold, otherwise this scenario is impossible.

If source >= target (aka, transferring all students from first part to second will do), then we mark source as a potential answer.

Otherwise, we have to use students from second carriage, so if additional >= (target-source), then we mark target as a potential answer.

These are all the cases we have to worry about and the total complexity is O(N log N) or O(N) if you're bothered with doing count sort.
→ Reply

3 years ago, # ^ | ☆ ← Rev. 2 +8

gen

The proof for the first statement. It is clear that we don't need to empty any carriage with more than 3 students. Also, emptying a carriage with 2 students and filling a carriage with 1 student is not optimal, since it is better to put that 1 student in the carraige with 2 guys. Thus there is such a border, and it is somewhere before the first carriage with more than 2 guys.

UPD: sorry, I misunderstood, it's only a proof for a maximum of 4 people in carriage.
→ Reply

3 years ago, # ^ | ☆ 0

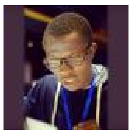
gen

Okay, the proof for the general statement is essentially the same. Suppose we empty a carriage with b people and fill a carriage with a people, $a < b$. It is clear that we better empty the carriage with a people, hence contradiction.
→ Reply

3 years ago, # | ☆ ← Rev. 2 0

Would someone point out why I'm getting a wrong answer on test 26 for this

submission 4808669 and 4799577 on 356B - Ксюша и Хемминг


Olayinka

submission [illegible] and [illegible] on 356B - Ксюша и Хемминг
→ Reply

mitstudent

3 years ago, # ^ | ☆ 0

Are you sure you are using 64-bit integers? (like long long)
→ Reply

Olayinka

3 years ago, # ^ | ☆ 0

yes, I used `unsigned long long`
→ Reply

TMandzu

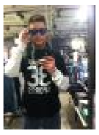
3 years ago, # | ☆ +3

In Problem "356D — Bags and Coins" , can we write masks in unsigned long long , so we can solve the problem in n*s/64 operations ?
→ Reply

kingofnumbers

3 years ago, # ^ | ☆ ← Rev. 2 0

But you should keep in mind that oprations on long long are slower than operations on int
→ Reply


TMandzu

3 years ago, # ^ | ☆ ← Rev. 2 0

Yes , I know time will be the same , but those bitmasks was needed on opencup contest , where 64 bits didn't passed (WA) , 32 passed , so I am asking what is wrong with 64 bit integers .
→ Reply

xiaodao

3 years ago, # | ☆ 0

There is a typo for problem E: $f(L) = 1^2 + 3^3$ ...
→ Reply

Swistakk

3 years ago, # | ☆ 0

Am I the only one who thinks that Compartments was an ugly problem? I got the wrong the wrong idea and I had to deal with eleven or thirteen cases and of course I had holes in my solution, but even when got the good idea this wasn't a nice problem. And editorial for this task is the best proof.
→ Reply

klamathix

3 years ago, # ^ | ☆ ← Rev. 2 +8

I'm not fond of it, but it's not really that bad. There is a solution in which you don't have any cases or choices to make. If you try to fix the number of compartments in the final solution, the number of 3-compartments and 4-compartments is uniquely determined. Looking at the input and the fixed candidate solution in sorted order you may have something like:

Say you fix the number of compartments to 5.

1 2 2 3 4 4

0 3 3 3 3 4

You just add the absolute difference for each position and then divide by 2. You can do that efficiently by using frequencies instead of values in about 3 — 4 lines of code.

It's the kind of problem that teaches you to keep things simple

It's the kind of problem that teaches you to keep things simple
(don't assume anything, jut try everything) which is a welcome
reminder from time to time.
→ Reply

Swistakk

3 years ago, # ^ | ☆ 0

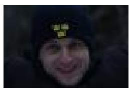
Yes, I know that solution (marcin_smu told me that) and I think that this is the nicest one. Indeed, I fell in trap of thinking "this can be done in any way!" and choose the wrong one, but as you can see, even author of this problem wasn't aware of that solution without any case-analysis.
→ Reply

apoorvj

3 years ago, # | ☆ ← Rev. 3 +2

Wrong tutorial? 357 B -FLAG DAY What if this is the test case n=6 m=3 dance1 1 2 3 dance2 1 4 5 dance3 4 2 6

the algorithm could fail on it.
→ Reply


Nerevar

3 years ago, # ^ | ☆ +11

Please, read the statement carefully before asking such questions. This case is not valid because in dance 3 we have two dancers (2 and 4), who had participated in some of the previous dances.
→ Reply

apoorvj

3 years ago, # ^ | ☆ +5

I am sorry, didnt get the problem statement. Now its clear. Thanks
→ Reply

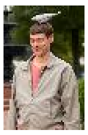
vijitthetopcoder

3 years ago, # | ☆ 0

In the second solution for Knight Tournament, if we set "next[cur] = cur + 1;", my understanding is we would always end up traversing each element, though we would process only the ones which are alive. Instead I think we should do

next[cur] = getNext(cur + 1); cur = next[cur];

Is my understanding correct or am I missing something here ?
→ Reply


jan25

18 months ago, # | ☆ 0

I'm trying to solve prob.A with dsu, also used path compression. Even though all test cases must use same amount of mem for my code, got MLE for 22nd case although n=100 http://ideone.com/MUWaHY
→ Reply

Esraa.Mohamed

14 months ago, # | ☆ 0

In Problem A. Group of Students :

I UnderStand All Statement Of This Problem But I Can't Explain Test 6

```
I'm Asking For Explain This Test

    Thanks , For Your Kind :)
```
→ Reply

6 months ago, # | ☆ 0

My solution for div1 A gives TLE, but approach is same as given in

editorial. Any idea why? Iam simply deleting elemnts in each match, so the editorial. Any idea why? I am simply deleting elemnts in each match, so the worst case complexity is O(n*log^2(n)), when each match has only 2 knights because code

→ Reply

FamIsProud