

OSAMA_ALKHODAIRY

BLOG

TEAMS

SUBMISSIONS

GROUPS

CONTESTS

PROBLEMSETTING

Osama_Alkhodairy's blog

Codeforces Round #613 (Div. 2) Editorial

By **Osama_Alkhodairy**, 3 days ago, , 

1285A - Mezo Playing Zoma

Let c_L and c_R be the number of 'L's and 'R's in the string respectively. Note that Zoma may end up at any integer point in the interval $[-c_L, c_R]$. So, the answer equals $c_R - (-c_L) + 1 = n + 1$.

code

```
#include <bits/stdc++.h>
using namespace std;
#define finish(x) return cout << x << endl, 0
#define ll long long

int n;
string s;

int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cin >> n >> s;
    cout << n + 1 << endl;
}
```

1285B - Just Eat It!

If there is at least a prefix or a suffix with non-positive sum, we can delete that prefix/suffix and end up with an array with sum \geq the sum of the whole array. So, if that's the case, the answer is "NO".

Otherwise, all the segments that Adel can choose will have sum $<$ than the sum of the whole array because the elements that are not in the segment will always have a strictly positive sum. So, in that case, the answer is "YES".

Time complexity: $O(n)$

code

```
#include <bits/stdc++.h>
using namespace std;
#define finish(x) return cout << x << endl, 0
#define ll long long

int n;
vector <int> a;

bool solve(){
    cin >> n;
    a.resize(n);
    for(auto &i : a) cin >> i;
    ll sum = 0;
    for(int i = 0 ; i < n ; i++){
        sum += a[i];
        if(sum <= 0) return 0;
    }
    sum = 0;
    for(int i = n - 1 ; i >= 0 ; i--){
        sum += a[i];
        if(sum <= 0) return 0;
    }
    return 1;
}

int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    int T;
    cin >> T;
    while(T--){
        if(solve()) cout << "YES\n";
        else cout << "NO\n";
    }
}
```

1285C - Fadi and LCM

There will always be a solution where a and b are coprime. To see why let's prime factorize a and b . If they share a prime factor we can omit all its occurrences from one of them, precisely from the one that has fewer occurrences of that prime. Now, let's prime factorize X . Since there will be at most 11 distinct primes, we can distribute them between a and b with a brute force.

For an easier implementation, you can loop over all divisors d of X , check if $LCM(d, \frac{X}{d})$ equals X , and minimize the answer with the pair $(d, \frac{X}{d})$.

Time complexity: $O(\sqrt{n})$

code

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

→ Pay attention

Before contest

[Educational Codeforces Round 80 \(Rated for Div. 2\)](#)

2 days

[Register now >](#)

→ Top rated

#	User	Rating
1	MiFaFaOvO	3520
2	tourist	3430
3	apiadu	3351
4	mnbvmar	3332
5	Benq	3290
6	LHiC	3276
7	TLE	3270
8	Radewoosh	3251
9	ecnerwala	3241
10	Um_nik	3240

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

→ Top contributors


#	User	Contrib.
1	antonlygubO_o	188
1	Errichto	188
3	tourist	176
4	Radewoosh	173
5	vovuh	166
6	pikmike	164
7	ko_osaga	163
8	Um_nik	160
9	majk	159
10	rng_58	155


[View all →](#)


→ Find user


Handle:


Find


- Recent actions
- chokudai** → [AtCoder Beginner Contest 151 Announcement](#) 


hongjun-7 → [Smallest Enclosing Circle/Sphere Problem](#) 


ko_osaga → [Hello 2020 Editorial](#) 


MikeMirzayanov → [About Canceled Codeforces Round #614 \(Div. 2\)](#) 


--Someone-- → [How hard is it to become red in CF](#) 


rpkv → [Can anyone please provide a simpler test case that my code fails ?](#) 


Osama_Alkhodairy → [Codeforces Round #613 \(Div. 2\) Editorial](#) 


Eddagdeg → [30 day challenge](#) 


BledDest → [Educational Codeforces Round 21 - Editorial](#) 


300iq → [Codeforces Round #612 - Editorial](#) 


rng_58 → [Dwango Programming Contest 6th Announcement \(The contest time is unusual!\)](#) 


tourist → [touriststream 001: SNWS 2020 R1](#) 


hmehta → [Topcoder SRM 774](#) 


geniucos → [Mentor the next generation of IOI medalists](#) 


chokudai → [AtCoder Beginner Contest 150 Announcement](#) 

LeiviniaBirdway → [Codeforces Round #597 \(Div. 2\) Editorial](#) 

MikeMirzayanov → [Technocup 2020 — Elimination Round 4 + Codeforces Round 606: Editorial](#) 

chokudai → [AtCoder Beginner Contest 149 Announcement](#) 

EmilConst → [IZhO 2020](#) 

S.Nakib → [Please provide some tutorial on component DP and interval DP.](#) 

```
#define finish(x) return cout << x << endl, 0
#define ll long long

ll x;

int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cin >> x;
    vector <ll> f;
    for(ll i = 2 ; i * i <= x ; i++){
        if(x % i == 0){
            ll cur = 1;
            while(x % i == 0){
                x /= i;
                cur *= i;
            }
            f.push_back(cur);
        }
    }
    if(x > 1) f.push_back(x);
    int n = f.size();
    ll ansa = 1e18, ansb = 1e18;
    for(int i = 0 ; i < (1 << n) ; i++){
        ll a = 1, b = 1;
        for(int j = 0 ; j < n ; j++){
            if((i >> j) & 1) a *= f[j];
            else b *= f[j];
        }
        if(max(a, b) < max(ansa, ansb)){
            ansa = a;
            ansb = b;
        }
    }
    cout << ansa << " " << ansb << endl;
}
```

easier implementation

```
#include <bits/stdc++.h>
using namespace std;
#define finish(x) return cout << x << endl, 0
#define ll long long

ll x;

ll lcm(ll a, ll b){
    return a / __gcd(a, b) * b;
}

int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    cin >> x;
    ll ans;
    for(ll i = 1 ; i * i <= x ; i++){
        if(x % i == 0 && lcm(i, x / i) == x){
            ans = i;
        }
    }
    cout << ans << " " << x / ans << endl;
}
```

1285D - Dr. Evil Underscores

We will solve this problem recursively starting from the most significant bit. Let's split the elements into two groups, one with the elements which have the current bit on and one with the elements which have the current bit off. If either group is empty, we can assign the current bit of X accordingly so that we have the current bit off in our answer, so we will just proceed to the next bit. Otherwise, both groups aren't empty, so whatever value we assign to the current bit of X , we will have this bit on in our answer. Now, to decide which value to assign to the current bit of X , we will solve the same problem recursively for each of the groups for the next bit; let ans_{on} and ans_{off} be the answers of the recursive calls for the on and the off groups respectively. Note that if we assign 1 to the current bit of X , the answer will be $2^i + ans_{off}$, and if we assign 0 to the current bit of X , the answer will be $2^i + ans_{on}$, where i is the current bit. So, simply we will choose the minimum of these two cases for our answer to be $2^i + \min(ans_{on}, ans_{off})$.

Time complexity: $O(n\log(maxa_i))$

code

```
#include <bits/stdc++.h>
using namespace std;
#define finish(x) return cout << x << endl, 0
#define ll long long

int n;
vector <int> a;

int solve(vector <int> &c, int bit){
    if(a.size() == 0 || bit < 0) return 0;
    vector <int> l, r;
    for(auto &i : c){
        if((i >> bit) & 1) == 0) l.push_back(i);
        else r.push_back(i);
    }
    if(l.size() == 0) return solve(r, bit - 1);
    if(r.size() == 0) return solve(l, bit - 1);
    return min(solve(l, bit - 1), solve(r, bit - 1)) + (1 << bit);
}

int main(){
```

d-agrawal → [Competitive Companion support added](#)

antontrygubO_o → [On problemsetting 2](#)

AI.Cash → [Efficient and easy segment trees](#)

huawei → [Huawei Honorcup Marathon 2](#)

BanazadehAria → [Strange Problem!](#)

[Detailed →](#)

```
ios_base::sync_with_stdio(0);
cin.tie(0);
cin >> n;
a.resize(n);
for(auto &i : a) cin >> i;
cout << solve(a, 30) << endl;
}
```

1285E - Delete a Segment

Ok, looking for a new number of segments in a union is actually hard. Let nw_i be the union of segments after erasing the i -th one. Obviously, each of the segments in $nw[i]$ has its left and right borders. Let me show you how to calculate the number of any of these two kinds. Let's choose left borders. I will call the set of left borders of the set s of segments lf_s .

Build the initial union of all segments (that is a standart algorithm, google it if you want). Call it $init$. We are asked to find $\max_i |nw[i]|$, but let's instead find $\max_i |nw[i]| - |init|$ (that is the difference of sizes of the initial union and the new one for i). Surely, adding $|init|$ to this value will be the answer. Moreover, $\max_i |nw[i]| - |init| = \max_i |lf_{nw[i]}| - |lf_{init}|$ and that's what we are going to calculate. Call that difference $diff_i$.

Let's do the following sweep line. Add queries of form $(l_i, i, 1)$ and $(r_i, i, -1)$. Process them in sorted order. Maintain the set of the open segments. This sweepline will add segment i on a query of the first type and remove segment i on a query of the second type. Initialize all the $diff_i$ with zeroes, this sweepline will help us to calculate all the values altogether.

Look at the all updates on the same coordinate x . The only case we care about is: the current set of open segments contain exactly one segment and there is at least one adding update. Let this currently open segment be j . Consider what happens with $nw[j]$. x is not in the lf_{init} because at least that segment j covers it. x is also in $lf_{nf[j]}$ because after erasing segment j x becomes a left border of some segment of the union (you are adding a segment with the left border x and points slightly to the left of x are no longer covered by segment j). Thus, $diff_j$ increases by 1.

The other possible cases are:

- there are no open segments currently — this is not important because x was a left border and stays as a left border;
- there are more than two open segments — not important because x will still be covered by at least one of them after erasing some other;
- there are no adding updates — x was a left border but doesn't become a new one.

Thus, we handled all the left border count increasing cases. But there are also a decreasing case. Left border can get removed if the segment you are erasing had its left border in the initial union and was the only segment with such left border. You can get lf_{init} while getting $init$. Then for each of lf_{init} you can count how many segments start in it. Finally, iterate over i and decrease $diff_i$ by one if the value for the left border of the segment i is exactly 1.

Finally, $diff_i$ is obtained, $(\max_i diff_i) + |init|$ is the answer.

Overall complexity: $O(n \log n)$.

Author: **MikeMirzayanov**
[code \(pikmike\)](#)

```
#include <bits/stdc++.h>

#define forn(i, n) for (int i = 0; i < int(n); i++)
#define x first
#define y second

using namespace std;

const int INF = 2e9;

typedef pair<int, int> pt;
map<int, int> ls;

int get(vector<pt> a){
    int cnt = 0;
    int l = -INF, r = -INF;
    sort(a.begin(), a.end());
    forn(i, a.size()){
        if (a[i].x > r){
            if (r != -INF)
                ls[l] = 0;
            ++cnt;
            l = a[i].x, r = a[i].y;
        }
        else{
            r = max(r, a[i].y);
        }
    }
    ls[l] = 0;
    return cnt;
}

void process(vector<pair<int, pt>> &qr, vector<int> &ans){
    set<int> open;
    forn(i, qr.size()){
        vector<int> op, cl;
        int j = i - 1;
        while (j + 1 < int(qr.size()) && qr[j + 1].x == qr[i].x){
            ++j;
            if (qr[j].y.x == 1)
                op.push_back(qr[j].y.y);
            else
                cl.push_back(qr[j].y.y);
        }
        if (open.size() == 1 && !op.empty()){
            ++ans[*open.begin()];
        }
        for (auto it : op){
```

```
        open.insert(it);
    }
    for (auto it : cl){
        open.erase(it);
    }
    i = j;
}

void solve(){
    int n;
    scanf("%d", &n);
    vector<pt> a(n);
    forn(i, n) scanf("%d%d", &a[i].x, &a[i].y);

    vector<pair<int, pt>> qr;
    forn(i, n){
        qr.push_back({a[i].x, {1, i}});
        qr.push_back({a[i].y, {-1, i}});
    }
    sort(qr.begin(), qr.end());

    vector<int> ans(n, 0);
    ls.clear();
    int cur = get(a);

    process(qr, ans);
    forn(i, n) if (ls.count(a[i].x)) ++ls[a[i].x];
    forn(i, n) if (ls[a[i].x] == 1) --ans[i];

    printf("%d\n", *max_element(ans.begin(), ans.end()) + cur);
}

int main(){
    int tc;
    scanf("%d", &tc);
    forn(i, tc)
        solve();
}
```

1285F - Classical?

Since $LCM(x, y) = \frac{x*y}{GCD(x,y)}$, it makes sense to try and fix $GCD(x, y)$. Let's call it g . Now, let's only care about the multiples of g in the input. Assume we divide them all by g . We now want the maximum product of 2 **coprime** numbers in this new array.

Let's sort the numbers and iterate from the biggest to the smallest, keeping a stack. Assume the current number you're iterating on is x . While there is a number in the stack coprime to x , you can actually pop the top of the stack; you'll never need it again. That's because this number together with a number smaller than x can never give a better product than that of a greater, or equal, number together with x ! Now, we just need to figure out whether there's a number coprime to x in the stack. This could be easily done with inclusion-exclusion. Assume the number of multiples of d in the stack is cnt_d ; the number of elements in the stack coprime to x is:

$$\sum_{d|x} \mu(d) * cnt_d$$

Where μ is the Mobius function. So we'll just iterate on the integers from greatest to smallest, and while there's a number coprime to x in the stack, we'll keep maximizing the answer, popping, and updating the array cnt . Then, we'll push x to the stack and also update cnt .

The complexity is $O(\sum_{i=1}^n \sigma_0(i)^2)$ where σ_0 is the divisor count function. That's because each number enters the routine of calculating the maximum product of a coprime pair σ_0 times, and we iterate through its divisors in this routine.

code (mohammedehab2002)

```
#include <bits/stdc++.h>
using namespace std;
#define MX 100000
int arr[100005], u[MX+5], cnt[MX+5];
vector<int> d[MX+5];
bool b[MX+5];
int coprime(int x)
{
    int ret=0;
    for (int i:d[x])
        ret+=cnt[i]*u[i];
    return ret;
}
void update(int x,int a)
{
    for (int i:d[x])
        cnt[i]+=a;
}
int main()
{
    for (int i=1;i<=MX;i++)
    {
        for (int j=i;j<=MX;j+=i)
            d[j].push_back(i);
        if (i==1)
            u[i]=1;
        else if ((i/d[i][1])%d[i][1]==0)
            u[i]=0;
        else
            u[i]=-u[i/d[i][1]];
    }
    int n;
    scanf("%d",&n);
```

```
long long ans=0;
for (int i=0;i<n;i++)
{
    int a;
    scanf("%d",&a);
    ans=max(ans,(long long)a);
    b[a]=1;
}
for (int g=1;g<=MX;g++)
{
    stack<int> s;
    for (int i=MX/g;i>0;i--)
    {
        if (!b[i*g])
            continue;
        int c=coprime(i);
        while (c)
        {
            if (__gcd(i,s.top())==1)
            {
                ans=max(ans,1LL*i*s.top()*g);
                c--;
            }
            update(s.top(),-1);
            s.pop();
        }
        update(i,1);
        s.push(i);
    }
    while (!s.empty())
    {
        update(s.top(),-1);
        s.pop();
    }
}
printf("%I64d",ans);
}
```

Tutorial of Codeforces Round #613 (Div. 2)

+115

[Osama_Alkhodairy](#)

3 days ago

[97](#)

Comments (97)

[Write comment?](#)

- 45 hours ago, <#> |

+19

thanks for the fast editorial :D

→ [Reply](#)
- 45 hours ago, <#> |

+11

Wow editorial even before system testing!!

→ [Reply](#)
- 45 hours ago, <#> |

← Rev. 2 **+4**

The problems were beautiful. Can someone please suggest some similar Project Euler questions to the F ? It definitely feels like a standard question.

P.S. — [Here](#) are my solutions to this lovely contest in case anybody wants to refer my code.

→ [Reply](#)
- 45 hours ago, <#> |

0

It's the first time I have seen editorial coming out before testing. Thanks :D

→ [Reply](#)
- 45 hours ago, <#> |

+4

<https://codeforces.com/contest/1285/submission/68548219>

why my solution fails on pretest 7? what is wrong in my appraoch.

→ [Reply](#)
- 44 hours ago, <#> [^](#) |

+3

You initial `11 ans = LONG_MAX`. If answer more than LONG_MAX you get WA. You can use LLONG_MAX

→ [Reply](#)
- 27 hours ago, <#> [^](#) |

0

Thanks bro it works.

→ [Reply](#)
- 44 hours ago, <#> |


0

can some help me with F classical problem

→ [Reply](#)
- 44 hours ago, <#> |

← Rev. 2 **-27**

The comment is hidden because of too negative feedback, click [here](#) to view it




44 hours ago, # |

▲ 0 ▼

Why tutorial is not available for problem E ?
→ [Reply](#).

sapjv




44 hours ago, # ^ |

▲ 0 ▼

I hope they are preparing a great editorial of problem E.
→ [Reply](#).

Anus1373




25 hours ago, # ^ |

▲ +16 ▼

Unfortunately they did not.
→ [Reply](#).

spookywooky




44 hours ago, # |

▲ +5 ▼

F accepted with a strange bitset solution [68535522](#)
→ [Reply](#).

FSTForces




44 hours ago, # |

▲ 0 ▼

For problem C: My Submission [68521316](#), I know I did some extra work still I feel that it should not give TLE. Can someone explain why it TLED on the very last case?

Test case 84: 963761198400
→ [Reply](#).

iamrk




43 hours ago, # ^ |

▲ +1 ▼

because it has 6720 factors and may perform bad on worse than n^2 TC
→ [Reply](#).

sumantopal07

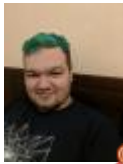


43 hours ago, # ^ |

▲ +1 ▼

Maybe this number has 6720 factors, which is much more than any cases before.
 $6720^2 \log(N)$ could cause TLE.
→ [Reply](#).

qxforever




43 hours ago, # |

▲ +24 ▼

Can you share who is the author of each problem?
→ [Reply](#).

Um_nik



43 hours ago, # ^ |

▲ +13 ▼

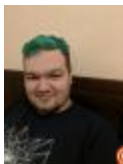
ABCD: me

E: **MikeMirzayanov**

F: **DeadPillow**

mohammedehab2002 is a solution author.
→ [Reply](#).

Osama_Alkhodairy




42 hours ago, # ^ |

▲ +3 ▼

OK, my guess was wrong.

Strange that nobody stopped you from using C. A was nice though.
→ [Reply](#).

Um_nik



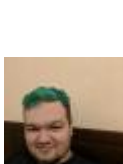
42 hours ago, # ^ |

▲ -17 ▼

What's wrong with C? Actually, in the beginning, we weren't aware of the easy implementation! The intended solution was the first one in the editorial.

Osama_Alkhodairy What was your guess though? :)
→ [Reply](#).

Osama_Alkhodairy




42 hours ago, # ^ |

▲ -10 ▼

I think I solved that one... 6 years ago? For the first time, I mean. Not sure that today was in first 5.

My guess was that C was added by **MikeMirzayanov** to "balance the difficulty".
→ [Reply](#).

Um_nik




42 hours ago, # ^ |

▲ -34 ▼

→ [Reply](#).

The comment is hidden because of too negative feedback, click [here](#) to view it

mohitsamant




43 hours ago, # |

▲ +2 ▼

For D, how does recursively solving for each of the groups ensure that we will reach an optimal answer?
→ [Reply](#).

chick_magnet



42 hours ago, # ^ |

← Rev. 3 ▲ +1 ▼

We are iterating over bits from higher to lower bit. Now it is known that $2^i > 2^{i-1} + 2^{i-2} + \dots + 1$

There can be two cases,

1. We will be able to get 0 on i th bit of our minimum value.
2. We will get 1 on i th bit of our minimum value, no matter what.

Just handle these cases.



Bojack

→ [Reply](#)



aayueshbar

41 hour(s) ago, <#> [^](#) | [▲](#) **+1** [▼](#)

But then how is the complexity of code not 2^n

→ [Reply](#)



Bojack

35 hours ago, <#> [^](#) | [▲](#) **+2** [▼](#)

The recursion depth won't be greater than $\log(a)$. Now notice that, when we handle the 2nd case, one might think that we will consider the same set of numbers twice. But it is not, because, these two cases are actually complementary set (Because if a number is in L set, it won't definitely be in R set), so we will end up considering each number at most $\log(a)$ times.

→ [Reply](#)



MrGary

28 hours ago, <#> [^](#) | [←](#) Rev. 2 [▲](#) **-9** [▼](#)

delete

→ [Reply](#)



8421BCD

11 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

1300+ now, how long it will take me to get 1600+?

→ [Reply](#)



MrGary

11 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

? what do you mean by saying it?

→ [Reply](#)



I_luv_Lana_Del_Rey

43 hours ago, <#> | [▲](#) **0** [▼](#)

In problem C how can they say there can be atmost 11 prime?

→ [Reply](#)



Bojack

42 hours ago, <#> [^](#) | [▲](#) **+8** [▼](#)

$2 * 3 * 5 * 7 * 11 * 13 * 17 * 19 * 23 * 29 * 31 * 37 > 10^12$

→ [Reply](#)



Doruletz

43 hours ago, <#> | [▲](#) **+8** [▼](#)

Why does the complexity at problem D is **$O(N \log(a))$** instead of $O(N * 2^{\log(a)}) = O(N * a)$?

→ [Reply](#)



Bojack

42 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

In the worst case, you will iterate over all numbers of array $\log(a)$ times.

→ [Reply](#)



Doruletz

25 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

Thanks

→ [Reply](#)



damjandzalev

42 hours ago, <#> | [←](#) Rev. 3 [▲](#) **0** [▼](#)

Problem B, 10, 10 5 -12 7 -10 20 30 -10 50 60

The maximum tastiness for Adel is 110, the sum of the array is 150, why is the answer No?

→ [Reply](#)



damjandzalev

42 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

Nevermind, I found my own mistake.

→ [Reply](#)



syed.al.mahi

22 hours ago, <#> [^](#) | [▲](#) **+1** [▼](#)

what was the mistake? I still don't know why the answer is No

→ [Reply](#)



damjandzalev

8 hours ago, <#> [^](#) | [▲](#) **0** [▼](#)

[20 30 -10 50 60] if you sum them up, they are 150 so Adel's max is 150. My mistake was, whenever my program encountered a negative number it wouldn't add them up and it would start over from 0. It should sum them up if the sum is greater than 0.

→ [Reply](#)



soul_voyage

42 hours ago, <#> | [▲](#) **0** [▼](#)

Solution without any difference array or binsearch — [68562620](#)

→ [Reply](#)



Abhinaviitbhu

42 hours ago, <#> | [▲](#) **0** [▼](#)

for(int i = 0 ; i < (1 << n) ; i++){ ll a = 1, b = 1; for(int j = 0 ; j < n ; j++){ if((i >> j) & 1) a *= f[j]; else b *= f[j]; } I know they are bitwise operator and left shift and right shift but what it is doing can someone tell

→ [Reply](#)



42 hours ago, <#> | [▲](#) **0** [▼](#)

please explain f classical problem

→ [Reply](#)

mohitsamant



tiktak2

42 hours ago, # | ▲ 0 ▼

Anyone can teach me or give me a link to understand brute force please? Thanks in advance.
→ [Reply](#).

42 hours ago, # | ▲ +32 ▼

Your F is easy to optimise to $\mathcal{O}(V \log V)$, where V is the upper bound on the input values.

For any input number a , we can add its divisors into our input numbers, as using a over them cannot make the LCA smaller. This is easy to do in $\mathcal{O}(V \log V)$ time.



mango_lassi

Then it suffices to find the two coprime numbers with maximum product, as if the optimal answer uses numbers a and b , it could just as well use the coprime numbers a and $\frac{b}{\gcd(a,b)}$ instead. Then we do not need to loop g , and doing what is described in the editorial for $g = 1$ takes $\mathcal{O}(\sum_{i=1}^n \sigma_0(i)) = \mathcal{O}(V \log V)$ time.

Submission: [68564248](#)
→ [Reply](#)

41 hour(s) ago, # ^ | ▲ +42 ▼

Okay this is so cool! I'll try to compile the other solutions I know here.

There's an $\mathcal{O}(V^2)$ idea: for each number a_i we can repeat the following: keep a list of all the numbers in the array, and take the largest element in the list; let's call it m . You can maximize the answer with $LCM(a_i, m)$, and then every multiple of $GCD(a_i, m)$ is useless because it can never give a better answer, so you can erase them from the list and repeat.

If instead of the list you keep a 0/1 array that tells you which elements exist in the list, you can optimize all these operations with bitset to achieve an $\mathcal{O}(\frac{V^2}{32})$ solution.



mohammedehab2002

Credits: [FSTForces](#)

There's also a good randomized solution based on [this trick](#). Choose a random permutation, and then the expected number of indices that increase the answer is $\mathcal{O}(\log(n))$. So, for each element in the permutation, if you can check whether it could increase the current answer and the check is positive, you can just try LCMing it with every single element in the array, since there are only $\mathcal{O}(\log(n))$ elements that will give a positive check! To perform this check, you can iterate through the divisors d of a_{p_i} , and then you want to check if there's an element a_j such that $\frac{a_{p_i} * a_j}{d} > ans$ and $GCD(a_{p_i}, a_j) = d$. Which is equivalent to $a_j > \frac{ans * d}{a_{p_i}}$ and $\frac{a_j}{d}$ is coprime with $\frac{a_{p_i}}{d}$. The editorial describes how to count the number of elements coprime to an element in an array. This problem is the same but for a suffix.

We also received a ton of heuristics that, together, are probably unbreakable.
→ [Reply](#)



priyanshugarg219

41 hour(s) ago, # | ▲ 0 ▼

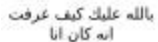
Is there anyone who has solved problem D with DP Please share your algo :)
→ [Reply](#)



archit_akg

39 hours ago, # | ▲ 0 ▼

any other approach for problem D ??
→ [Reply](#)



It_Wasnt_Me

38 hours ago, # ^ | ▲ +5 ▼

You can solve it using trie :)
→ [Reply](#)

38 hours ago, # ^ | ← Rev. 2 ▲ 0 ▼

Nice! One question, maybe it's because I don't know enough about trie. Do you know why people call the function "search" (example below is `solve`) with `search(1, 30)` or `search(0, 29)` ?



MysD

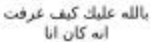
Example: [Code](#)

I don't understand why the guys are using 29 (or 30) and not any other number. I think this is kind of Level of the tree. Can you help?

Thanks!
→ [Reply](#)

36 hours ago, # ^ | ▲ +1 ▼

Because the max number in the array will be $(2^{30} - 1)$



It_Wasnt_Me

and in the trie we traverse on bits so we have maximum 30 bits to traverse on it, think about binary representation.
→ [Reply](#)



MysD

24 hours ago, # ^ | ▲ 0 ▼

I get it now. Thank you!
→ [Reply](#)



lzhang

39 hours ago, # | ▲ +10 ▼

Not understanding the editorial of E, can some one elaborate more on the approach? thanks!
→ [Reply](#)

- kai824

34 hours ago, # 1 |

+6

Yes, I also don't really understand the solution after the part about building the initial union of all segments.

Any help is much appreciated thanks.

→ Reply.
- frank3215

34 hours ago, # 2 | ← Rev. 2


+6

I was also a little confused before I read the code.

Instead of building the initial union, the code `get()` s all the right borders of the union, `ls` , and the initial number of segments, `cur` .

And while you `process()` the intervals(i.e. scanning them), you check if there is there is exactly a open segment x before the current right segment, and if there is, `++ans[x]` | (the difference if you remove x). Otherwise, the current right border would not appear in any answer.

Finally, you `solve()` the question by checking if removing segment x will remove a right border in `ls` , then adding `ans[x]` to `cur` .


→ Reply.
- 

lzhang

7 hours ago, # 3 |

0

Thanks!


→ Reply.
- 

Ayers

38 hours ago, # 4 |

0

Can anyone find the mistake in my code? I got the wrong answer on test 63 and I have no idea how to fix it. 68545474

→ Reply.
- 

madlogic

37 hours ago, # 5 | ← Rev. 3

0

Please correct me if I'm wrong but in the code for D above

Isn't the line

```
if (l.size() == 0) return solve(r, bit - 1)
```

meant to be

```
if (l.size() == 0) return solve(r, bit - 1) + (1 << bit)
```

EDIT: Sorry, I misunderstood the problem statement I thought we were supposed to find the value of X.

→ Reply.
- sh_maestro

35 hours ago, # 6 |

0

Thanks for an excellent set of problems and a timely well-written editorial!

→ Reply.

Hd7

34 hours ago, # 7 |

0

In problem C, why are there at most 11 distinct primes?

→ Reply.

gupta_samarth

34 hours ago, # 8 1 |

0

You should read other comments first, to see if your query has already been answered. [This comment](#).

Read more about [primorials](#).

→ Reply.

Hd7

34 hours ago, # 9 1 |

0

thank u, It's very nice.

→ Reply.


destroyerguy

34 hours ago, # 10 |

0

Can anyone suggest some other problems similar to Problem D?

→ Reply.



KhaledFarhat

23 hours ago, # 11 1 |

0

[What are some good xor problems on codeforces?](#)

[Xor and String classic problems](#)

[A Beautiful Technique for Some XOR Related Problems](#)

→ Reply.


mayankk_agarwal

19 hours ago, # 12 1 | ← Rev. 2

0

An interesting follow up can be finding the value of x. As I understand this should be straightforward.

→ Reply.



akshay1998

28 hours ago, # 13 1 |


0

I didn't understand the part where we said that bit will be 1 if both groups will be non-empty. Will anyone give me an example to understand that part of the problem D

→ Reply.

9 of 12

12/01/20, 7:35 pm



shahriaroo

25 hours ago, # ^ |


+3

Consider 5,6,7 the answer is 2.
The 3rd bit in all of them is set, therefore, we can set the 3rd bit in X to make sure that the 3rd bit in the answer is 0.

The 2nd bit is set in 6 and 7 but not in 5 so no matter we set it in X or not it will be set in the answer because when we XOR it with them there will be a difference between X and at least one of them.

Now 5 is in one group and 6,7 are in another. In the group that consists of only 5 we have the 1st bit set to 1, therefore, we set the 1st bit in X to 1. So X should be either 5 or 7 and the answer is 2.

→ [Reply](#).



priyanshugarg219


25 hours ago, # ^ |

← Rev. 2

0

If both groups are non-empty then we perform 3rd return statement and look carefully there is addition of 2^{**i} for ith bit

→ [Reply](#).



DarkFloyd

25 hours ago, # ^ |

0


<< from editorial : **If both groups aren't empty, whatever value we assign to the current bit of X, we will have this bit on in our answer.** >>

Let n=2, two numbers are 9(1001) and 5(0101) . And we are considering the 3rd bit (counted from 0). On the value of x ..

if we on that bit .. (so x=1000 now) after xor two numbers will be 1(0001) and 13(1101), ans=13 if we on that bit .. (so x=0000 now) after xor two numbers will be 9(1001) and 5(0101) , ans=9

the 3rd bit is ON in either cases . Hope u got it.

→ [Reply](#).




akshay1998

23 hours ago, # ^ |

0

Thanks for such great explanation.

→ [Reply](#).




MrGary

27 hours ago, # |

+3

For problem F ,I used an amazing greedy algorithm ,of course it is wrong ,But why I got TLE instead of WrongAnswer? <https://codeforces.com/contest/1285/submission/68568056>

→ [Reply](#).




MrGary

27 hours ago, # ^ |

+4

ok... , I only let $3000=2750$ it got WA instead of TLE.

→ [Reply](#).




GaryMr

27 hours ago, # ^ |

0

Be careful ,man .Don't expect to pass a problem without the proper solution.

→ [Reply](#).



sahil.kalamkar


27 hours ago, # |

← Rev. 2

0

Can someone tell me why this solution is getting timed out for question C?
<https://codeforces.com/contest/1285/submission/68565564>

→ [Reply](#).




SkySurfer

27 hours ago, # ^ |

+5

In for loop use long long instead of integer the variable is overflowing and it is giving TLE

→ [Reply](#).




sahil.kalamkar

25 hours ago, # ^ |

0

Thanks! It worked. I really feel stupid now.:(

→ [Reply](#).



gulshan_1997


26 hours ago, # |

0

1) <https://codeforces.com/contest/1285/submission/68534255> 2) <https://codeforces.com/contest/1285/submission/68585814>

Why the 1st code gives wrong ans for test 7 while the 2nd code works fine ?
(only difference between the two is initial value of ans, LONG_MAX doesn't work while $10^{*}12$ works)

→ [Reply](#).




amit_839

25 hours ago, # ^ |

+7

Value of LONG_MAX is equivalent to INT_MAX. You should have used LLONG_MAX. Try to print all of them and compare.

→ [Reply](#).



grand_zeno


26 hours ago, # |

← Rev. 2

0

Can someone explain how the complexity for problem F is derived?Won't we be looping over a number's divisors multiple times each time it enters the routine to calculate the maximum product of two coprime numbers (to find if there still is a number coprime to it in the stack)?

→ [Reply](#).



SkySurfer

25 hours ago, # |

0

Can someone explain me the time complexity of problem D ...I came up with the similar approach but was not able to find the time complexity !!!TIA

→ [Reply](#).

25 hours ago, # ^ |

+8



shahriaroo

In each iteration, we are iterating over the array and splitting the array into two subarrays so if the size of the left subarray in the i th iteration is b_i the recurrence relation in the depth of i will be $T(n) = T(b_i) + T(n - b_i) + cn$. Now if you draw the recursion tree, you'll see that the sum of all $T(n)$ on the same depth is $O(n)$. The depth of the tree is $\log(\max(a_i))$ which is less than 31 and the running time is $O(n\log(\max(a_i)))$.

→ Reply



benedict_

24 hours ago, # |

0

I dont understand solution 1285B what editorial coder is doing is checking the sum if less than zero or not at any point of time...but how abt case 7 4 3 -1 answer should be No.. Right??

→ Reply

17 hours ago, # ^ |

← Rev. 4 0

Yes the answer should be no, as adel could pick $7+4+3=14>13$ That's why the author calculated the cummlative sum in reverse order as well

A simple mathematical proof could be as follows:

consider:

1. sum => sum of the whole array
2. prefix[i] => sum of the array from index 0 to i
3. suffix[j] => sum of the array from index n-1 to n-1-j
4. sum[i:j] => sum of the array from index i to index j

then

- $\text{sum}[i:j] = \text{sum} - (\text{prefix}[i-1] + \text{suffix}[n-j-2])$ if $i>0$ and $j<n-1$

or

- $\text{sum}[i:j] = \text{sum} - \text{prefix}[i-1]$ if $j==n-1$

or

- $\text{sum}[i:j] = \text{sum} - \text{suffix}[n-j-2]$ if $i==0$

then from these equations, u can see that if any cummlative prefix sum or cummlative suffix sum is less than or equal zero then

$\text{sum}[i:j]>\text{sum}$ and the answer is NO

otherwise

$\text{sum}[i:j]<\text{sum}$ and the answer is YES

→ Reply



benedict_

15 hours ago, # ^ |

0

Wow Man really thankyou....There isn't any good work than helping a beginner. Thankyou so much. I got the logic there.

→ Reply



jerry_30

24 hours ago, # |

+5

define finish(x) return cout << x << endl, 0

what does this line do?

→ Reply



geranazavr555

22 hours ago, # ^ |

← Rev. 2 0

It creates the alias for printing a single value x to stdout and exiting with exit-code 0. For ex, finish("test") will print "test" and terminate program. It's not used in solutions above.

→ Reply



Meiji

23 hours ago, # |

+11

Can someone explain me F? I didn't quite get it

→ Reply



prav1609

22 hours ago, # |

0

In problem B, why my solution is giving WA? My Submission:<https://codeforces.com/contest/1285/submission/68610213> I simply made a segment tree and excluding the node which contains the whole interval [1,n],I checked for all other node if they have $\text{sum}>=\text{sum}$ of all elements..

→ Reply



MylnikovNikolay

22 hours ago, # ^ |

← Rev. 2 0

{3 -1 3 -1} breaks your solution, because you check all (not all, some of them) segments with length equal to power of two, but answer is [3 -1 3]

→ Reply



prav1609

19 hours ago, # ^ |

+5

Ok ..now I get it...silly of me not to think of the cases..once again thanks a lot!!!

→ Reply




_pratik_shukla_

21 hour(s) ago, # |

0

can someone tell what will be the output for problem B for case 9 9 9 -1 9 9 9 and why because im not able to connect with the editorial and the problem

→ Reply




20 hours ago, # |

what will happen in 2nd question if the first and last elements of array is zero

→ [Reply](#)

udayps055




19 hours ago, # ^ |

Answer comes out to be "NO" because the other guy can just select the interval [2,n-1] and get sum=sum of all elements.

→ [Reply](#)

prav1609




19 hours ago, # ^ |

another similar approach use maximum sum subarray concept on a[1:len(a)-1] and a[:len(a)] and take the max of the two and max(a[0],a[len(a)-1])

→ [Reply](#)

dukaandaar




19 hours ago, # |

just a general question guys. do you think that the level of this round was lower than other div2 rounds? just wanted to know so as to analyze my performance. by the way thanks to the authors and editorial writers for the round

→ [Reply](#)

dukaandaar



18 hours ago, # |

This is an alternate approach for E:

Let us store the segments sorted (first by start, then by end) in $seg[1...n]$

Define $p(i)$: Number of segments in union of segments $1, ..., i$

Define $me(i)$: $max\{seg[1].end, ..., seg[i].end\}$

Obtaining both of these is not a difficult task and can be performed in $O(n)$

Now, we process the segments in order: **n down to 2** while maintaining a set called V whose definition is: When we start processing segment i , V contains the union of segments $i+1...n$ **in sorted order** (V contains segments which are union of $seg[i+1]...seg[n]$). When we start processing segment n, V is empty. **It must be clear that at any point V contains segments which are pairwise disjoint.**

To calculate the number of segments in union if segment i is removed, find the number of segments in V whose start is greater than $me(i-1)$. This can be achieved by binary search. Let this number be x. Update answer as $max(answer, p(i-1) + x)$. The reason for this is: From segments $1...i-1$ the max end point is $me(i-1)$. This covers all segments from V which have $start \leq me(i-1)$. The rest of the segments in V are x in number.


To make it clear, we first process segment n (and update V to include segment n) then process segment n-1 (again update V to include segment n-1) and so on. We do not process segment 1. For segment 1, the answer is simply the size of V after processing segments $2...n$.

Updating V: After processing segment i, we need to update V. Let $(l, r) = seg[i]$ and segments in V be $(l_1, r_1), ..., (l_k, r_k)$. It must be clear that the segments in V are pairwise disjoint. Also, $l \leq l_1 \leq l_2 \dots \leq l_k$. To add (l, r) to V, keep removing segments from beginning whose $l_i \leq r$ (and while doing so, update $r = max(r, r_i)$). Finally you will have (l, r) such that it is disjoint from each segment in V. Add this (l, r) to beginning of V.

I have omitted some minor details.

→ [Reply](#)

xvenom99




14 hours ago, # |

Whoops, sorry, guys, I reread my editorial for E and it seems that I was too tired to write anything :) There are some nice comments detailing it, but I updated mine anyway. Hopefully, now it's easier to get what was going in my head when I wrote the solution. The part with sweepline still sounds pretty complicated but I guess I can't explain it better without doing an entire lecture on what sweepline actually is. The code should be pretty clear, refer to it maybe.

→ [Reply](#)

pikmike




2 hours ago, # |

In problem D, I'm having trouble understanding, why we split the integers into groups according to the MSB. I understand, that if there any 2 bits different, the answer will have bit 1 at that place but what's the thought process behind splitting them into groups for the remaining bits?

→ [Reply](#)

FusionX



76 minutes ago, # |

A solution for [1285D - Dr. Evil Underscores](#) with trie with an idea like the editorial but more understandable [68656872](#)

→ [Reply](#)

jorgeajl

