


[AWOO](#) [BLOG](#) [TEAMS](#) [SUBMISSIONS](#) [GROUPS](#) [CONTESTS](#) [PROBLEMSETTING](#)

awoo's blog

Educational Codeforces Round 115 Editorial

By [awoo](#), [history](#), 10 months ago, translation, 

1598A - Computer Game

Idea: [BledDest](#)
[Tutorial](#)

1598A - Computer Game

At first glance, it seems like a graph problem. And indeed, this problem can be solved by explicitly building a graph considering cells as the vertices and checking that there is a safe path from start to finish via DFS/BFS/DSU/any other graph algorithm or data structure you know. But there's a much simpler solution.

Since there are only two rows in a matrix, it's possible to move from any cell in the column i to any cell in column $i + 1$ (if they are both safe, of course). It means that as long as there is at least one safe cell in each column, it is possible to reach any column of the matrix (and the cell $(2, n)$ as well).

It's easy to see that if this condition is not met, there exists a column with two unsafe cells — and this also means that this column and columns to the right of it are unreachable. So, the problem is reduced to checking if there is a column without any unsafe cells. To implement this, you can read both rows of the matrix as strings (let these strings be s_1 and s_2) and check that there is a position i such that both $s_1[i]$ and $s_2[i]$ are equal to 1.

[Solution \(BledDest\)](#)

```
def solve():
    n = int(input())
    s1 = input()
    s2 = input()
    bad = False
    for i in range(n):
        bad |= s1[i] == '1' and s2[i] == '1'
    if bad:
        print('NO')
    else:
        print('YES')

t = int(input())
for i in range(t):
    solve()
```

1598B - Groups

Idea: [fcspartakm](#)
[Tutorial](#)

1598B - Groups


Since there are only five days, we can iterate over the two of them that will be the answer.

Now we have fixed a pair of days a and b and want to check if it can be the answer.

→ Pay attention

Before contest
[Codeforces Round #811 \(Div. 3\)](#)
05:17:39
[Register now »](#)

→ don_vanchos

 Rating: **1184**
Contribution: **0**



don_vanchos

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

→ Top rated

#	User	Rating
1	tourist	3771
2	jiangly	3688
3	Um_nik	3539
4	slime	3498
5	djqu_cpp	3486
6	MiracleFaFa	3466
7	ksun48	3452
8	Radewoosh	3406
9	greenheadstrange	3393
10	xtqqwq	3382

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

→ Top contributors

#	User	Contrib.
1	awoo	182
1	-is-this-fft-	182
3	YouKn0wWho	177
4	Um_nik	175
5	Monogon	172
5	dario2994	172
7	antontrygubO_o	168
7	maroonrk	168
9	adamant	167
10	errorgorn	163

[View all →](#)

→ Find user

Handle:

Find

All students can be divided into four groups: marked neither of days a and b , marked only day a , marked only day b and marked both days.

Obviously, if the first group is non-empty, days a and b can't be the answer.

Let's call the number of students, who only marked day a , cnt_a and the number of students, who only marked day b , cnt_b .

If either of cnt_a or cnt_b exceed $\frac{n}{2}$, then days a and b can't be the answer as well. Otherwise, we can always choose $\frac{n}{2} - cnt_a$ students from the ones who marked both days and send them to day a . The rest of the students can go to day b .

Solution (BledDest)

```
t = int(input())
for i in range(t):
    n = int(input())
    a = [[] for i in range(n)]
    for j in range(n):
        a[j] = list(map(int, input().split()))
    ans = False
    for j in range(5):
        for k in range(5):
            if k != j:
                cnt1 = 0
                cnt2 = 0
                cntno = 0
                for z in range(n):
                    if a[z][j] == 1:
                        cnt1 += 1
                    if a[z][k] == 1:
                        cnt2 += 1
                    if a[z][j] == 0 and a[z][k] == 0:
                        cntno += 1
                if cnt1 >= n // 2 and cnt2 >= n // 2 and cntno == 0:
                    ans = True
    if ans:
        print('YES')
    else:
        print('NO')
```

1598C - Delete Two Elements

Idea: [fcspartakm](#)

[Tutorial](#)

1598C - Delete Two Elements

First of all, instead of the mathematic mean, let's consider the sum of elements. If the mathematic mean is k , then the sum of elements of the array is $k \cdot n$. Let's denote the sum of elements in the original array as s . Note s is always an integer.

If we remove two elements from the array, the resulting sum of elements should become $k \cdot (n - 2) = \frac{s \cdot (n - 2)}{n}$. So, the sum of the elements we remove should be exactly $\frac{2s}{n}$.

If $\frac{2s}{n}$ is not an integer, the answer is 0 (to check that, you can simply compare $(2s) \bmod n$ with 0). Otherwise, we have to find the number of pairs (i, j) such that $i < j$ and $a_i + a_j = \frac{2s}{n}$. This is a well-known problem.

To solve it, you can calculate the number of occurrences of each element and store it in some associative data structure (for example, `map` in C++). Let cnt_x be the number of occurrences of element x . Then, you should iterate on the element a_i you want to remove and check how many elements match it, that is, how many elements give exactly $\frac{2s}{n}$ if you add a_i to them. The number of these elements is just $cnt_{\frac{2s}{n} - a_i}$. Let's sum up all these values for every element in the array.

→ Recent actions

[Pranjaliko](#) → [Why codeforces is not accepting this code when all the outputs are matching correctly?](#)

[anonymous112233](#) → [HELP NEEDED: HARD RANGE QUERY PROBLEM](#)

[awoo](#) → [Educational Codeforces Round 132 Editorial](#)

[ScorpioDagger](#) → [Finally Cyan! UPD: Now, BLUE!](#)

[Vladosiya](#) → [Codeforces Round #811 \(Div. 3\)](#)

[WhyAsh5114](#) → [ZCO eligibility for students choosing non-conventional but valid education \(India\)](#)

[AAK](#) → [Indian ICPC 2022 Regionals — Discussion](#)

[Cirno_9baka](#) → [CodeTON Round 2 Editorial](#)

[QAQAutoMaton](#) → [IOI2022 China Team](#)

[AquaMoon](#) → [CodeTON Round 2](#)

[Ghosted](#) → [Maybe Codeforces needs a more complete anti cheating system](#)

[Qingyu](#) → [XXII Open Cup, Grand Prix of China \(EC-Finals\)](#)

[chokudai](#) → [AtCoder Beginner Contest 262 Announcement](#)

[hemant_thakur](#) → [Help needed in DP problem](#)

[Yubai](#) → [An Interesting combination problem](#)

[jli505](#) → [TeamsCode Summer 2022 Contest](#)

[Iftekhar_Hakim_K](#) → [Invitation to Replay of BUET Inter University Programming Contest 2022](#)

[marinyordanov](#) → [IATI 2018 \(ban?\) \(Note that this is my new profile and therefore i am newbie :\)\)](#)

[scipianus](#) → [Codeforces Round #271 \(Div. 2\) Editorial](#)

[SyadouHayami](#) → [Ask for the solution of CF1704F](#)

[abhinav700](#) → [problem while declaring a 3d array](#)

[dsbdsb](#) → [fighting for rating 2200](#)

[RDFZzzx](#) → [A brief introduction of Segment Tree\(I\):Build tree and query without update](#)

[1900_mashups](#) → [Mashups for Aspiring Masters.](#)

[MikeMirzayanov](#) → [Rule about third-party code is changing](#)

[Detailed →](#)

Unfortunately, this sum is not the answer yet. We need to take care of two things:

- if for some index i , $2 \cdot a_i = \frac{2s}{n}$, then a_i matches itself, so you have to subtract the number of such elements from the answer;
- every pair of elements is counted twice: the first time when we consider the first element of the pair, and the second time — when we consider the second element of the pair. So, don't forget to divide the answer by 2.

Solution (Neon)

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

using namespace std;

int main() {
    int t;
    scanf("%d", &t);
    while (t--) {
        int n;
        scanf("%d", &n);
        vector<int> a(n);
        map<int, int> cnt;
        for (auto &x : a) {
            scanf("%d", &x);
            cnt[x] += 1;
        }
        long long sum = accumulate(a.begin(), a.end(), 0LL);
        if ((2 * sum) % n != 0) {
            puts("0");
            continue;
        }
        long long need = (2 * sum) / n;
        long long ans = 0;
        for (int i = 0; i < n; ++i) {
            int a1 = a[i];
            int a2 = need - a1;
            if (cnt.count(a2)) ans += cnt[a2];
            if (a1 == a2) ans -= 1;
        }
        printf("%lld\n", ans / 2);
    }
}
```

1598D - Training Session

Idea: **BledDest**

Tutorial

1598D - Training Session

There are many different ways to solve this problem, but, in my opinion, the easiest one is to count all possible triples and subtract the number of bad triples.

The first part is easy — the number of ways to choose 3 elements out of n is just $\frac{n \cdot (n-1) \cdot (n-2)}{6}$. The second part is a bit tricky.

What does it mean that the conditions in the statements are not fulfilled? There is a pair of problems with equal difficulty, and there is a pair of problems with the same topic. Since all problems in the input are different, it means that every bad triple has the following form: $[(x_a, y_a), (x_b, y_a), (x_a, y_b)]$ — i. e. there exists a problem such that it shares the difficulty with one of the other two problems, and the topic with the remaining problem of the triple.

This observation allows us to calculate the number of bad triples as follows: we will iterate on the "central" problem (the one that shares the topic with the second problem and the

difficulty with the third problem). If we pick (x_a, y_a) as the "central" problem, we need to choose the other two. Counting ways to choose the other problems is easy if we precalculate the number of problems for each topic/difficulty: let $cntT_x$ be the number of problems with topic x , and $cntD_y$ be the number of problems with difficulty y ; then, if we pick the problem (x, y) as the "central one", there are $cntT_x - 1$ ways to choose a problem that shares the topic with it, and $cntD_y - 1$ ways to choose a problem that has the same difficulty — so, we have to subtract $(cntT_x - 1)(cntD_y - 1)$ from the answer for every problem (x, y) .

Solution (Neon)

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

using namespace std;

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    int t;
    cin >> t;
    while (t--) {
        int n;
        cin >> n;
        vector<int> a(n), b(n), ca(n + 1), cb(n + 1);
        for (int i = 0; i < n; ++i) {
            cin >> a[i] >> b[i];
            ca[a[i]]++; cb[b[i]]++;
        }
        long long ans = n * 1LL * (n - 1) * (n - 2) / 6;
        for (int i = 0; i < n; ++i)
            ans -= (ca[a[i]] - 1) * 1LL * (cb[b[i]] - 1);
        cout << ans << '\n';
    }
}
```

1598E - Staircases

Idea: **BledDest**

Tutorial

1598E - Staircases

The solution consist of two main parts: calculate the initial number of staircases and recalculate the number of staircases on query.

The constraints were pretty loose, so we'll do the first part in $O(nm)$ and the second part in $O(n + m)$ per query.

However, it's worth mentioning that faster is possible. The first part can surely be done in $O(n + m)$ and can probably be done in $O(1)$. The second part can be done in $O(\log n)$ per query.

It's important to notice is that the only staircase that satisfy the requirements for both types is the staircase that consists of a single cell. Thus, staircases of both types can be calculated almost separately.

Let's define "base" staircases as the staircases that can't be prolonged further in any direction. There are $O(n + m)$ of them on the grid.

If a staircase consists of at least two cells, it's a part of exactly one *base* staircase. At the same time, every segment of a *base* staircase is a valid staircase by itself.

Thus, the main idea of calculating the initial answer is the following. Isolate each *base* staircase and determine its length len (possibly, in $O(n + m)$). Add $\binom{len-1}{2}$ (the number of segments of length at least 2) to the answer. Add extra nm one cell staircases afterwards.

If you draw the *base* staircases on the grid, you can easily determine their starting cell. The *base* staircases, that start by going one cell to the right, start from the first row. The *base* staircases, that start by going one cell to the bottom, start from the first column. Notice that both types can start from cell $(1, 1)$.

The updates can be handled the following way. The answer always changes by the number of staircases that pass through cell (x, y) (if you ignore its state). If the cell becomes free, then these staircases are added to the answer. Otherwise, they are subtracted from it.

That can be calculated for two cases as well. Go first down, then right, as far as possible. Let it be cnt_1 steps. Go first left, then up, as far as possible. Let it be cnt_2 steps. Then $cnt_1 \cdot cnt_2$ staircases are added to the answer. Then change the order of steps in both directions to calculate the other type of staircases. Beware of one cell staircases again.

To achieve $O(n + m)$ for precalc, you can calculate the length of each *base* staircase with a formula. To achieve $O(\log n)$ per query, you can first enumerate cells in each *base* staircase separately, then maintain the set of segments of adjacent free cells in it.

Solution (awoo)

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

#define forn(i, n) for (int i = 0; i < int(n); i++)

using namespace std;

int main() {
    int n, m, q;
    scanf("%d%d%d", &n, &m, &q);
    vector<vector<int>> a(n, vector<int>(m, 1));
    long long ans = 0;
    forn(x, n) forn(y, m){
        if (x == 0){
            for (int k = 1;; ++k){
                int nx = x + k / 2;
                int ny = y + (k + 1) / 2;
                if (nx == n || ny == m) break;
                ans += k;
            }
        }
        if (y == 0){
            for (int k = 1;; ++k){
                int nx = x + (k + 1) / 2;
                int ny = y + k / 2;
                if (nx == n || ny == m) break;
                ans += k;
            }
        }
    }
    ans += n * m;
    forn(i, q){
        int x, y;
        scanf("%d%d", &x, &y);
        --x, --y;
        forn(c, 2){
            int l = 1, r = 1;
            while (true){
                int nx = x + (l + c) / 2;
                int ny = y + (l + !c) / 2;
                if (nx == n || ny == m || a[nx][ny] == 0)
                    break;
                ++l;
            }
            while (true){
                int nx = x - (r + !c) / 2;
```



```

        int ny = y - (r + c) / 2;
        if (nx < 0 || ny < 0 || a[nx][ny] == 0) break;
        ++r;
    }
    if (a[x][y] == 0){
        ans += 1 * r;
    }
    else{
        ans -= 1 * r;
    }
}
ans += a[x][y];
a[x][y] ^= 1;
ans -= a[x][y];
printf("%lld\n", ans);
}
}

```

1598F - RBS

Idea: **BledDest**Tutorial

1598F - RBS

The constraint $n \leq 20$ is a clear hint that we need some exponential solution. Of course, we cannot try all $n!$ permutations. Let's instead try to design a solution with bitmask dynamic programming.

A string is an RBS if its balance (the difference between the number of opening and closing brackets) is 0, and the balance of its each prefix is non-negative. So, let's introduce the following dynamic programming: $dp_{m,b,f}$ is the greatest number of RBS-prefixes of a string if we considered a mask m of strings s_i , the current balance of the prefix is b , and f is a flag that denotes whether there already has been a prefix with negative balance. We can already get rid of one of the states: the current balance is uniquely determined by the mask m . So, this dynamic programming will have $O(2^{n+1})$ states.

To perform transitions, we need to find a way to recalculate the value of f and the answer if we append a new string at the end of the current one. Unfortunately, it's too slow to simply simulate the process. Instead, for every string s_i , let's precalculate the value $go(s_i, f, x)$ — how does the flag and the answer change, if the current flag is f , and the current balance is x .

The resulting flag will be `true` in one of the following two cases:

- it is already `true`;
- the string we append creates a new prefix with non-positive balance.

The second case can be checked as follows: let's precalculate the minimum balance of a prefix of s_i ; let it be c . If $x + c < 0$, the flag will be `true`.

Calculating how the answer changes is a bit trickier. If the current flag is already `true`, the answer doesn't change. But if it is `false`, the answer will increase by the number of new RBS-prefixes. If the balance before adding the string s_i is b , then we get a new RBS-prefix for every prefix of s_i such that:

- its balance is exactly $(-b)$ (to compensate the balance we already have);
- there is no prefix with balance $(-b - 1)$ in s_i before this prefix.

To quickly get the number of prefixes meeting these constraints, we can create a data structure that stores the following information: for every balance j , store a sorted vector of positions in s_i with balance equal to j . Then, to calculate the number of prefixes meeting the constraints, we can find the first position in s_i with balance equal to $(-b - 1)$ by looking at the beginning of the vector for $(-b - 1)$, and then get the number of elements less than this one from the vector for balance $(-b)$ by binary search.

These optimizations yield a solution in $O(2^n \log A + A \log A)$, although it's possible to improve to $O(2^n + A \log A)$ if you precalculate each value of $go(s_i, f, x)$ for every string s_i .

Solution (BledDest)

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

using namespace std;

const int INF = int(1e9);

const int N = 20;
const int M = (1 << N);

struct BracketSeqn
{
    int balance;
    int lowestBalance;
    vector<int> queryAns;

    pair<int, bool> go(int x, bool f)
    {
        if(f)
            return make_pair(0, true);
        else
            return make_pair(x < queryAns.size() ? queryAns[x] : 0, x +
lowestBalance < 0);
    }

    BracketSeqn() {};
    BracketSeqn(string s)
    {
        vector<int> bal;
        int cur = 0;
        int n = s.size();
        for(auto x : s)
        {
            if(x == '(')
                cur++;
            else
                cur--;
            bal.push_back(cur);
        }
        balance = bal.back();
        lowestBalance = min(0, *min_element(bal.begin(), bal.end()));
        vector<vector<int>> negPos(-lowestBalance + 1);
        for(int i = 0; i < n; i++)
        {
            if(bal[i] > 0) continue;
            negPos[-bal[i]].push_back(i);
        }
        queryAns.resize(-lowestBalance + 1);
        for(int i = 0; i < queryAns.size(); i++)
        {
            int lastPos = int(1e9);
            if(i != -lowestBalance)
                lastPos = negPos[i + 1][0];
            queryAns[i] = lower_bound(negPos[i].begin(),
negPos[i].end(), lastPos) - negPos[i].begin();
        }
    };
};
```



```

int dp[M][2];
char buf[M];
int total_bal[M];

int main()
{
    int n;
    scanf("%d", &n);
    vector<BracketSeqn> bs;
    for(int i = 0; i < n; i++)
    {
        scanf("%s", buf);
        string s = buf;
        bs.push_back(BracketSeqn(s));
    }
    for(int i = 0; i < (1 << n); i++)
        for(int j = 0; j < n; j++)
            if(i & (1 << j))
                total_bal[i] += bs[j].balance;
    for(int i = 0; i < (1 << n); i++)
        for(int j = 0; j < 2; j++)
            dp[i][j] = -int(1e9);
    dp[0][0] = 0;
    for(int i = 0; i < (1 << n); i++)
        for(int f = 0; f < 2; f++)
        {
            if(dp[i][f] < 0) continue;
            for(int k = 0; k < n; k++)
            {
                if(i & (1 << k)) continue;
                pair<int, bool> res = bs[k].go(total_bal[i],
                f);
                dp[i ^ (1 << k)][res.second] = max(dp[i ^ (1
                << k)][res.second], dp[i][f] + res.first);
            }
        }
    printf("%d\n", max(dp[(1 << n) - 1][0], dp[(1 << n) - 1][1]));
}

```

1598G - The Sum of Good Numbers

Idea: **BledDest**

Tutorial

1598G - The Sum of Good Numbers

Let's denote a as the largest of the terms of the sum, and b is the smaller one.

Consider 2 cases: $|a| = |x| - 1$ or $|a| = |x|$.

If $|a| = |x| - 1$, then $|b| = |x| - 1$. So we need to find two consecutive substrings of length $|x| - 1$ such that if we convert these substrings into integers, their sum is equal to x .

If $|a| = |x|$, let lcp be the largest common prefix of a and x if we consider them as strings. Then $|b| = |x| - lcp$ or $|b| = |x| - lcp - 1$. So it is necessary to check only these two cases, and whether b goes before or after a (in the string s).

Thus, we have reduced the number of variants where the substrings for a and b are located to $O(n)$. It remains to consider how to quickly check whether the selected substrings are suitable. To do this, you can use hashes (preferably with several random modules).

Solution (Neon)


```

#include <bits/stdc++.h>

using namespace std;

#define forn(i, n) for (int i = 0; i < int(n); ++i)

const int MOD[] = { 597804841, 618557587, 998244353 };
const int N = 500 * 1000 + 13;
const int K = 3;

using hs = array<int, K>;

int add(int x, int y, int mod) {
    x += y;
    if (x >= mod) x -= mod;
    if (x < 0) x += mod;
    return x;
}

int mul(int x, int y, int mod) {
    return x * 1LL * y % mod;
}

hs get(const int& x) {
    hs c;
    forn(i, K) c[i] = x;
    return c;
}

hs operator +(const hs& a, const hs& b) {
    hs c;
    forn(i, K) c[i] = add(a[i], b[i], MOD[i]);
    return c;
}

hs operator -(const hs& a, const hs& b) {
    hs c;
    forn(i, K) c[i] = add(a[i], -b[i], MOD[i]);
    return c;
}

hs operator *(const hs& a, const hs& b) {
    hs c;
    forn(i, K) c[i] = mul(a[i], b[i], MOD[i]);
    return c;
}

int n, m;
string s, sx;
hs sum[N], pw[N];

hs get(int l, int r) {
    return sum[r] - sum[l] * pw[r - l];
}

vector<int> zfunction(const string& s) {
    int n = s.size();
    vector<int> z(n);
    int l = 0, r = 0;
    for (int i = 1; i < n; ++i) {
        if (i <= r) z[i] = min(z[i - l], r - i + 1);
        while (i + z[i] < n && s[z[i]] == s[i + z[i]])
            ++z[i];
    }
}

```



```

    if (i + z[i] - 1 > r) {
        l = i;
        r = i + z[i] - 1;
    }
}
return z;
}

int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);

    cin >> s >> sx;
    n = s.size();
    m = sx.size();

    pw[0] = get(1);
    forn(i, N - 1) pw[i + 1] = pw[i] * get(10);
    sum[0] = get(0);
    forn(i, n) sum[i + 1] = sum[i] * get(10) + get(s[i] - '0');
    hs x = get(0);
    forn(i, m) x = x * get(10) + get(sx[i] - '0');

    if (m > 1) for (int i = 0; i + 2 * (m - 1) <= n; ++i) {
        if (get(i, i + m - 1) + get(i + m - 1, i + 2 * (m - 1)) == x) {
            cout << i + 1 << ' ' << i + m - 1 << '\n';
            cout << i + m << ' ' << i + 2 * (m - 1) << '\n';
            return 0;
        }
    }

    auto z = zfunction(sx + "#" + s);

    for (int i = 0; i + m <= n; ++i) {
        int lcp = z[m + i + 1];
        for (int len = m - lcp - 1; len <= m - lcp; ++len) {
            if (len < 1) continue;
            if (i + m + len <= n && get(i, i + m) + get(i + m, i + m + len) == x) {
                cout << i + 1 << ' ' << i + m << '\n';
                cout << i + m + 1 << ' ' << i + m + len << '\n';
                return 0;
            }
            if (i >= len && get(i - len, i) + get(i, i + m) == x) {
                cout << i - len + 1 << ' ' << i << '\n';
                cout << i + 1 << ' ' << i + m << '\n';
                return 0;
            }
        }
    }

    assert(false);
}

```



Tutorial of Educational Codeforces Round 115 (Rated for Div. 2)

+114

[awoo](#)

10 months ago

72



Comments (72)

[Write comment?](#)

10 months ago, # | ☆

← Rev. 4

+11

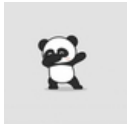
The easiest solution for C(in my opinion): 131432277

P.S. $sum * (n-2) = (sum - a[l] - a[r]) * n \Rightarrow$ we can find for every $a[l]$ all $a[r]$ in



t0t_samij_lv3n0k

1. $\sum_{i=1}^n (i-2) = (\sum_{i=1}^n i) - 2n = \frac{n(n+1)}{2} - 2n$ \Rightarrow we can find for every $a[i]$ an $a[j]$ in $O(n \log n)$ using binary search
 → [Reply](#)



MoonKnight.

10 months ago, # ^ | ☆

+9

If you simplify that equation, you would get $(a[i] + a[j]) = (2 * \text{sum}) / n$, so basically this is just a 2-sum problem where the target value is $(2 * \text{sum}) / n$,
 → [Reply](#)



t0t_samij_lv3n0k

10 months ago, # ^ | ☆

0

Yep, u r right (:

→ [Reply](#)

BadassCoder

10 months ago, # ^ | ☆

0

Can you please check my soln. I have used 2-sum approach but it is giving me WA on test 2.

131563369

→ [Reply](#)

10 months ago, # ^ | ☆

0

++



online_engineer_2024

I also tried the same way and getting wa on 78th in 2tc..MoonKnight. can you pls tell me why it's wa.

solution: 131653903

→ [Reply](#)

aryanc403

10 months ago, # ^ | ☆ 0

I havent looked at the correctness of your algorithm but
`ans = ans + c * d;` line
 has int overflow for sure.

→ [Reply](#)

10 months ago, # ^ | ☆ 0



online_engineer_2024

XD. Thanks, for mentioning this.
 now, after using it long it still showing wa on that tc. Can you please tell me why is it so?

→ [Reply](#)

aryanc403

10 months ago, # ^ | ☆ 0

Changing
`ans = ans + c * d;`
 to `ans = 1LL*ans + c * d;`
 doesn't fixes anything.
`ans` is already
`long long int`. Over flow is in multiplication

multiplication
of `c*d` ,
change this
to `ans =`
`ans +`
`c*1LL*d` .

Also `ll`
`val = (n *`
`(n - 1)) /`
`2;` has int
overflow.
Just having
`val` of
type `long`
`long int`
doesn't do
anything.
`n` and `n-`
`1` are still of
`int` type.
Change this
to `n*1LL*`
`(n-1)/2` .

Also, next
time just
uses C++ 64
bit and
`long`
`long`
everywhere.

Also, `if`
`(st.size()`
`== 1)` is
incorrect fix.
You will still
get WA on
Spoiler
→ Reply



10 0
months
ago,
^
| ☆



online_engineer_2024
Thanks.
Got my
mistake.
→
Reply

10 months ago, # ^ | ☆ ▲ 0 ▼



I-dont-know-FFT
also you are using a unordered_map here
which is giving TLE in test case 13, so
better use map only . you can read this
article (its quite cool)
[https://codeforces.com/blog/entry/62393?](https://codeforces.com/blog/entry/62393?#comment-464775)
#comment-464775
→ Reply



t0t_samij_iv3n0k

9 months ago, Rev. 2 | ☆ 0 ▼
OR... You can use ur own hash
function (:
→ Reply

10 months ago, # | ☆

← Rev. 3 ▲ +20 ▼



paulzrm

Sorry I forgot that x is a good number :_-

→ [Reply](#)



aopo

10 months ago, # | ☆

▲ +7 ▼

Video Editorial of E (Staircases)

→ [Reply](#)



killswitch_200

10 months ago, # | ☆

▲ 0 ▼

IDk why but my solution for C gives TLE on Test 17 even though it's the exact same solution as the tester's :

<https://codeforces.com/contest/1598/submission/131542518>

→ [Reply](#)



parth_sak12

10 months ago, # ^ | ☆

▲ 0 ▼

Don't know why, but using `unordered_map` gives TLE in this question I have the same problem... use `map` instead this will solve problem.

→ [Reply](#)



Namit

10 months ago, # ^ | ☆

▲ +1 ▼

Maybe because you're using `unordered_map`?

[Link to Another CF Blog.](#)

→ [Reply](#)



killswitch_200

10 months ago, # ^ | ☆

▲ +3 ▼

Thnxx a lot!!

→ [Reply](#)



tot_samij_lv3n0k

10 months ago, # ^ | ☆

▲ 0 ▼

Yep, i fixed ur solution: [131548931](#)

P.S. check this:

<https://codeforces.com/blog/entry/62393>

→ [Reply](#)

10 months ago, # | ☆

▲ 0 ▼

Why using `unordered_map` gives TLE in this question... searching elements using hash map should be faster than the map which takes $\log(n)$ time to search instead.

[131514464](#)

Also I don't get it... how by adding just these **2 lines** which where on a blog,

```
mp.reserve(1024);
mp.max_load_factor(0.25);
```

solves the problem of TLE and my soln gets accepted.

[131524838](#)



parth_sak12

Then I tried a **custom hash** function rather than the built-in hash function. Which also got **accepted**. Also the same solution gets **accepted** in **python using dictionaries** which also uses hashmap.

Does this mean that the C++ built-in hash function is not good ?

[131520347](#)

I tried to find but didn't get some concrete answers when to use `map` or `unordered_map` ?

Or using custom hash is better not ?

Please if anyone can help it in.

→ [Reply](#)

10 months ago, # ^ | ☆

▲ 0 ▼

Yeah, you should use custom hash!

Fixed solution: [131549476](#)



t0t_samij_lv3n0k

Check this: <https://codeforces.com/blog/entry/62393>

→ [Reply](#)

10 months ago, # ^ | ☆

▲ 0 ▼

There are many hacks on unordered_map and I suppose we should avoid using it in Codeforces contests. Here a [blog](#) written by neal showing how it works.

→ [Reply](#)



YangTY



Thallium54

10 months ago, # ^ | ☆

▲ 0 ▼

It's just because c++'s hash function is deterministic so people can hack on this. `max_load_factor()` basically increase the number of buckets which will break the malicious hacks.

→ [Reply](#)

10 months ago, # ^ | ☆

▲ 0 ▼

As you mentioned that solution with python dictionary got accepted and Thallium54 suggessted in reply that c++'s hash function is deterministic, so I searched for python's and found this [here](#), python uses a hash function which is deterministic within a single run.

Also the custom hash function which you are using in c++ solution is also deterministic within a single run.

→ [Reply](#)



ankit4129



CM_Dan4Life

10 months ago, # ^ | ☆

▲ 0 ▼

Note to self: ALWAYS use custom hash function when dealing with unordered map no matter how unnecessary in a codeforces contest

→ [Reply](#)

10 months ago, # ^ | ☆

▲ +16 ▼

The funny thing in this contest is **sjcsjc** 's template had custom_hash but still he got hacked for using c++ default hash. [131404435](#) After that he dropped from 4th rank to 87th rank among rated users.

→ [Reply](#)



keep__calm



sjcsjc

10 months ago, # ^ | ☆

▲ +1 ▼

I am a noob.

→ [Reply](#)



sjcsjcsjc

10 months ago, # ^ | ☆

▲ +16 ▼

He is a noob.

→ [Reply](#)

10 months ago, # ^ | ☆

▲ +18 ▼

```
mp.reserve(1024);
mp.max_load_factor(0.25);
```

only changes the test that's required to hack the solution, it doesn't make it unhackable.

→ [Reply](#)



dorijanlendvaj

10 months ago, # | ☆

← Rev. 4

▲ 0 ▼

They seemed to be difficult during the contest, but now, I think them is not really tricky. Why I have feelings like this? How can I deal with it?

→ [Reply](#)



CY0531

10 months ago, # | ☆

▲ 0 ▼

Nice solution for D1



LMXZ

Need solution for D:

→ [Reply](#)



YangTY

10 months ago, # | ☆

▲ +13 ▼

I suppose that the time complexity of F should be $O(n2^n + A \log A)$ instead of $O(2^n + A \log A)$. (:

Could someone tell me if there are some mistakes?

→ [Reply](#)



Mazaalai

10 months ago, # ^ | ☆

▲ 0 ▼

Can you tell me what is A in this time complexity? I understand where N and 2^N comes from but I just can't find anything related to A.

→ [Reply](#)



aniervs

10 months ago, # ^ | ☆

▲ 0 ▼

You are right, it's $n \cdot 2^n$ instead of 2^n .

→ [Reply](#)



vantaablackk

10 months ago, # | ☆

▲ +3 ▼

Any other solution for problem D?

→ [Reply](#)



Osmosis

10 months ago, # | ☆

▲ 0 ▼

In Problem D, I don't understand why the only bad triples are (xa,ya),(xb,ya), (xa,yb). Isn't there a possibility of having three same topics, such (1, 1), (1, 2), (1, 3)? I've been looking at the explanation for so long but still can't figure it out :(

→ [Reply](#)



YangTY

10 months ago, # ^ | ☆

▲ 0 ▼

but their difficulties are different so they aren't bad triples.

The problem statement says "**at least one of the conditions are met**"

→ [Reply](#)



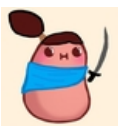
Osmosis

10 months ago, # ^ | ☆

▲ 0 ▼

Ah, now I get it. I really appreciate your reply. Thanks!!!

→ [Reply](#)



Karabasan

10 months ago, # | ☆

← Rev. 3 ▲ -9 ▼

Spoiler

→ [Reply](#)



TudorStefan

10 months ago, # | ☆

← Rev. 2 ▲ 0 ▼

For D, can someone explain to me why there can't be a triple of the form [(Xa, Ya), (Xb, Yb), (Xa, Yb)]? EDIT: I figured it out, it's actually the same thing except that the third pair is the central one, instead of the first.

→ [Reply](#)



Watermelon

10 months ago, # ^ | ☆

▲ 0 ▼

Yes, correct. I had the same doubt.

→ [Reply](#)

10 months ago, # | ☆

▲ 0 ▼

For 1508C:



toraoh

for 10000.

No, you don't have to use map in such a hacky way (or in another word, full of patches).

131572801

=====

Instead of keeping a map of all elements, we keep a map for number of **first i items** ([0, i-1])

So the core logic can be much more easier:

in C#:

```
for (int i = 0; i < n; i++){
    // update the answer
    if (cnt.ContainsKey(target - data[i])) answer += cnt[target - data[i]];
    // update the cnt map, for (i + 1) to use.
    if (!cnt.ContainsKey(data[i])) cnt.Add(data[i], 0);
    cnt[data[i]]++;
}
```

or in C++:

```
for (int i = 0; i < n; i++){
    // update the answer
    answer += cnt[target - data[i]];
    // update the cnt map, for (i + 1) to use.
    cnt[data[i]]++;
}
```

→ [Reply](#)



Shady.ELharakly

10 months ago, # | ☆

← Rev. 12 ▲ 0 ▼

i tried to add photo for my comment but the comment become empty any one can help me

→ [Reply](#)



devanshmishra111

10 months ago, # | ☆

▲ 0 ▼

In **problem D** can anyone please explain why we won't be overcounting in the editorial solution?

→ [Reply](#)



_Body

10 months ago, # ^ | ☆

▲ 0 ▼

Because he told you in the problem statement that the problems are distinct

→ [Reply](#)



Aoi_Hill

10 months ago, # ^ | ☆

▲ 0 ▼

problems and diff. are unique pair, no overlap when multiplication as in editorial

→ [Reply](#)



hack_learn

10 months ago, # | ☆

▲ 0 ▼

Regarding problem D, I have a question, and I want a simple answer The first is where the values in this equation $(n - 1) * (n - 2) / 6$ come from. I know it's about knowing the number of ways to choose, but I don't know where the division by six came from and why we subtracted it from one and two

→ [Reply](#)

10 months ago, # ^ | ☆

▲ 0 ▼

You can have an overview of basic combinatorics. It is like NC3 that



mohit_joshi

You can have an overview of basic combinatorics. It is like $N \cdot (N-1) \cdot (N-2)$, that means you have to select any three out of N so when we apply this, it becomes $N! / (3! \cdot (N-3)!)$, you can expand $N!$ as $N \cdot (N-1) \cdot (N-2) \cdot (N-3) \cdot (N-4) \dots$ and so on, you can see that $(N-3)!$ which is in denominator will cancel the numerator from $(N-3) \cdot (N-4) \cdot (N-5) \dots$, and that last you will be left with $(N \cdot (N-1) \cdot (N-2)) / 3!$ and $3! = 3 \cdot 2 \cdot 1 = 6$, so at last it becomes $N \cdot (N-1) \cdot (N-2) / 6$.

→ Reply

10 months ago, # | ☆

▲ +8 ▼

Alternative solution for 1598D - Training Session. I did see restriction that all problems pairs of parameters are different, but I had no idea how to use this fact. My solution doesn't rely on this fact at all. Solution turned out to be much harder so I wasted a lot of time to implement it during round.



r57shell

Here is main idea. Think of how to calculate all triples with different topics.

There is easy way

Apply the same idea to problem's difficulties. We count something twice. What to do?

Well

→ Reply

10 months ago, # | ☆

▲ +8 ▼

Detailed description how to implement 1598E - Staircases in $O((n + m + q) \log(n + m))$ time and space.

Solution

→ Reply



r57shell



Pawabhi

10 months ago, # | ☆

▲ 0 ▼

for que B <https://codeforces.com/contest/1598/submission/131729850> can u check this once i don't which test i am missing so pls

→ Reply

10 months ago, # ^ | ☆

← Rev. 2

▲ 0 ▼

```
if(r>=n/2&&q>=n/2&&s==0)
```

```
{
```

```
cout<<"YES"<<endl;
```

```
y=1; break;//you forget a break here
```

```
}
```

→ Reply



Amit7777



Pawabhi

10 months ago, # ^ | ☆

▲ 0 ▼

ok thanks bro

→ Reply



r57shell

10 months ago, # | ☆

▲ +17 ▼

A slightly different solution of 1598F - RBS.

Solution

→ Reply



Zaid_25

10 months ago, # | ☆

▲ 0 ▼

in problem C why i am getting tle.. Pls tell[problem:1598C]
<https://codeforces.com/contest/1598/submission/131512968>

→ Reply



daily_solver

7 months ago, # ^ | ☆

▲ 0 ▼

This is due to the use of `unordered_map` instead of `map`. Try `map` and you will get AC.

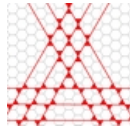
→ Reply

→ [Reply](#)

tushar_kumar

new, 3 months ago, # ^ | ☆ ← Rev. 2 ▲ 0 ▼

even , i used unordered map earlier it also gave me tle but when i am using map it is not giving tle . can anyone explain why is it so ? please.

→ [Reply](#)

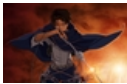
nonrice

new, 5 weeks ago, # ^ | ☆ Rev. 4 ▲ 0 ▼

I think it may be because of special anti-hash testcases.

Unordered map relies on hashing to determine where the value of a key is stored. When there are hash collisions, the the hash function makes the collided keys share the same position. This is done with a linked list, which needs to be linearly searched if you are looking for a key that has collided with other keys.

Essentially, you can design testcases that abuse this workaround, causing `unordered_map` access complexity to approach `O(n)`

→ [Reply](#)

CaptLevi

9 months ago, # | ☆ ▲ 0 ▼

132438878 can anybody help me with this I'm getting tle even though the time complexity of my son is same as that of the solution :)

→ [Reply](#)

CaptLevi

9 months ago, # ^ | ☆ ▲ 0 ▼

^in problem c

→ [Reply](#)

Bulgroz

9 months ago, # ^ | ☆ ▲ 0 ▼

This may upset you, but unordered_map uses more time than normal map. Confer to this link web :

<https://codeforces.com/blog/entry/62393?#comment-464775>

→ [Reply](#)

CaptLevi

9 months ago, # ^ | ☆ ▲ 0 ▼

thnx man

→ [Reply](#)

veerendra5057

9 months ago, # | ☆ ▲ 0 ▼

133831334 i use python yo solve this que .. when i tested my code on my computer it gave correct ans for the given example .. but when i submit .. it says runtime error everytime . someone plz help me

→ [Reply](#)

munditva

8 months ago, # | ☆ ▲ 0 ▼

My solution for F fails at test 32. Can someone please look at the [submission](#) and point out the error?

I would be happy to explain what my code does if it is not clear. Thanks.

→ [Reply](#)

CM_Dan4Life

7 months ago, # | ☆ ▲ 0 ▼

A Dp approach to problem E in $O(NM + Q * \min(N, M))$

[Solution](#)

→ [Reply](#)

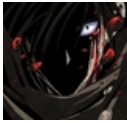


vibhavrathhe

7 months ago, # | ☆

▲ 0 ▼

Good job bro!

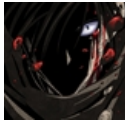
→ [Reply](#)

mkmeden

7 months ago, # | ☆

▲ 0 ▼

<https://codeforces.com/problemset/submission/1598/141426355> Can someone help , why my code gave TLE on testcase 11 of problem C

→ [Reply](#)

mkmeden

7 months ago, # | ☆

▲ 0 ▼

fixed

→ [Reply](#)

daily_solver

7 months ago, # | ☆

▲ 0 ▼

In C, I used unordered_map and it gave me TLE in test case 19th. I tried map and it gave me AC. Can somebody explain why?

→ [Reply](#)

chetan_2002

6 months ago, # | ☆

▲ 0 ▼

In the groups question, is it possible for the same student to attend lectures in both groups?

→ [Reply](#)

tgp07

6 months ago, # | ☆

▲ 0 ▼

For problem D, is there anyway to find the number of triples that are different both in topic and in difficulty?

→ [Reply](#)

DeepSpace_85

4 months ago, # | ☆

▲ 0 ▼

I am not able to get why TLE using Java in Question C, even though my Time Complexity is $O(n)$152782475

→ [Reply](#)

vito_t

new, 6 weeks ago, # | ☆

▲ 0 ▼

Problem A can be solved using DFS like this: 161393885.

→ [Reply](#)