


Codeforces Round #690 (Div. 3)

[A. Favorite Sequence]


CODE FORCES
 Sponsored by Telegram

Dyf3244 | Logout

[HOME](#) [TOP](#) [CONTESTS](#) [GYM](#) [PROBLEMSET](#) [GROUPS](#) [RATING](#) [EDU](#) [API](#) [CALENDAR](#) [HELP](#) [ABOUT](#) [STATUS](#)

[PROBLEMS](#) [SUBMIT CODE](#) [MY SUBMISSIONS](#) [STATUS](#) [HACKS](#) [STANDINGS](#) [CUSTOM INVOCATION](#)

A. Favorite Sequence

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp has a favorite sequence $a[1 \dots n]$ consisting of n integers. He wrote it out on the whiteboard as follows:

- he wrote the number a_1 to the left side (at the beginning of the whiteboard);
- he wrote the number a_2 to the right side (at the end of the whiteboard);
- then as far to the left as possible (but to the right from a_1), he wrote the number a_3 ;
- then as far to the right as possible (but to the left from a_2), he wrote the number a_4 ;
- Polycarp continued to act as well, until he wrote out the entire sequence on the whiteboard.

a_1 a_3 ... a_4 a_2

The beginning of the result looks like this (of course, if $n \geq 4$).

For example, if $n = 7$ and $a = [3, 1, 4, 1, 5, 9, 2]$, then Polycarp will write a sequence on the whiteboard $[3, 4, 5, 2, 9, 1, 1]$.
You saw the sequence written on the whiteboard and now you want to restore Polycarp's favorite sequence.

Input
The first line contains a single positive integer t ($1 \leq t \leq 300$) — the number of test cases in the test. Then t test cases follow.
The first line of each test case contains an integer n ($1 \leq n \leq 300$) — the length of the sequence written on the whiteboard.
The next line contains n integers b_1, b_2, \dots, b_n ($1 \leq b_i \leq 10^9$) — the sequence written on the whiteboard.

Output
Output t answers to the test cases. Each answer — is a sequence a that Polycarp wrote out on the whiteboard.

Example

input

```
6
7
3 4 5 2 9 1 1
4
9 2 7 1
11
8 4 3 1 2 7 8 7 9 4 2
1
42
2
11 7
8
1 1 1 1 1 1 1
```

output

```
3 1 4 1 5 9 2
9 1 2 7
8 2 4 4 3 9 1 7 2 8 7
42
11 7
1 1 1 1 1 1 1
```

Note
In the first test case, the sequence a matches the sequence from the statement. The whiteboard states after each step look like this:
 $[3] \Rightarrow [3, 1] \Rightarrow [3, 4, 1] \Rightarrow [3, 4, 1, 1] \Rightarrow [3, 4, 5, 1, 1] \Rightarrow [3, 4, 5, 9, 1, 1] \Rightarrow [3, 4, 5, 2, 9, 1, 1]$.

Codeforces Round #690 (Div. 3)

[Finished](#)

[Practice](#)

★

→ [Virtual participation](#)

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

[Start virtual contest](#)

→ [Practice](#)

You are registered for practice. You can solve problems unofficially. Results can be found in the contest status and in the bottom of standings.

→ [Clone Contest to Mashup](#)

You can done this contest to a mashup.

[Clone Contest](#)

→ [Submit?](#)

Language: GNU G++17 7.3.0

Choose file: [浏览...](#) 未选择文件.

[Submit](#)

→ [Last submissions](#)

Submission	Time	Verdict
101264174	Dec/15/2020 17:41	Accepted

→ [Problem tags](#)

[implementation](#) [two pointers](#)

No tag edit access

→ [Contest materials](#)

- Announcement

Codeforces (c) Copyright 2010-2020 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Dec/16/2020 18:07:46^{UTC+8} (g1).
Desktop version, switch to [mobile version](#).
[Privacy Policy](#)

Supported by




- 题目大意：给定一个输入规律是 $a_1, a_3, \dots, a_4, a_2$ 的数列，输出原数列
- 模拟

- 把给定的数列前 $\frac{n+1}{2}$ 个位置放到答案的奇数位置上，把后面的位置放到答案的偶数位置上即可

```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

int a[maxn];
int ans[maxn];
int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int n ; cin >> n;
        for ( int i = 1 ; i <= n ; i++ ) cin >> a[i];
        for ( int i = 1 ; i <= (n+1)/2 ; i++ )
            ans[2*i-1] = a[i];
        for ( int i = n ; i > (n+1)/2 ; i-- )
            ans[(n-i+1)*2] = a[i];
        for ( int i = 1 ; i <= n ; i++ )
            cout << ans[i] << " ";
        cout << endl;
    }
}
```

[B. Last Year's Substring]


CODE FORCES
 Sponsored by Telegram

DYF3244 | [Logout](#)

[HOME](#) [TOP](#) [CONTESTS](#) [GYM](#) [PROBLEMSET](#) [GROUPS](#) [RATING](#) [EDU](#) [API](#) [CALENDAR](#) [HELP](#) [ABOUT](#) [STATUS](#)

[PROBLEMS](#) [SUBMIT CODE](#) [MY SUBMISSIONS](#) [STATUS](#) [HACKS](#) [STANDINGS](#) [CUSTOM INVOCATION](#)

B. Last Year's Substring

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp has a string $s[1 \dots n]$ of length n consisting of decimal digits. Polycarp performs the following operation with the string s **no more than once** (i.e. he can perform operation 0 or 1 time):

- Polycarp selects two numbers i and j ($1 \leq i \leq j \leq n$) and removes characters from the s string at the positions $i, i+1, i+2, \dots, j$ (i.e. removes substring $s[i \dots j]$). More formally, Polycarp turns the string s into the string $s_1 s_2 \dots s_{i-1} s_{j+1} s_{j+2} \dots s_n$.

For example, the string $s = "20192020"$ Polycarp can turn into strings:

- "2020" (in this case $(i, j) = (3, 6)$ or $(i, j) = (1, 4)$);
- "2019220" (in this case $(i, j) = (6, 6)$);
- "020" (in this case $(i, j) = (1, 5)$);
- other operations are also possible, only a few of them are listed above.

Polycarp likes the string "2020" very much, so he is wondering if it is possible to turn the string s into a string "2020" in no more than one operation? Note that you can perform zero operations.

Input
The first line contains a positive integer t ($1 \leq t \leq 1000$) — number of test cases in the test. Then t test cases follow.

The first line of each test case contains an integer n ($4 \leq n \leq 200$) — length of the string s . The next line contains a string s of length n consisting of decimal digits. It is allowed that the string s starts with digit 0.

Output
For each test case, output on a separate line:

- "YES" if Polycarp can turn the string s into a string "2020" in no more than one operation (i.e. he can perform 0 or 1 operation);
- "NO" otherwise.

You may print every letter of "YES" and "NO" in any case you want (so, for example, the strings yEs, yes, Yes and YES will all be recognized as positive answer).

Example

input

```
6
8
20192020
8
22019020
4
2020
5
20002
6
729040
6
200200
```

output

```
YES
YES
YES
NO
NO
NO
NO
```

Note
In the first test case, Polycarp could choose $i = 3$ and $j = 6$.

In the second test case, Polycarp could choose $i = 2$ and $j = 5$.

In the third test case, Polycarp did not perform any operations with the string.

Codeforces Round #690 (Div. 3)
[Finished](#)
[Practice](#)


[Virtual participation](#)
 Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.
[Start virtual contest](#)

[Practice](#)
 You are registered for practice. You can solve problems unofficially. Results can be found in the contest status and in the bottom of standings.

[Clone Contest to Mashup](#)
 You can clone this contest to a mashup.
[Clone Contest](#)

[Submit?](#)
 Language: GNU G++17 7.3.0
 Choose file: 浏览... 未选择文件.
[Submit](#)

[Last submissions](#)

Submission	Time	Verdict
101274733	Dec/15/2020 17:51	Accepted

[Problem tags](#)
[implementation](#) [strings](#) No tag edit access

[Contest materials](#)

- Announcement

Codeforces (c) Copyright 2010-2020 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Dec/16/2020 18:08:04 UTC+8 (g1).
Desktop version, switch to [mobile version](#).
[Privacy Policy](#)

Supported by



- 题目大意：求问给定一个字符串当中，能够通过至多一次删除若干个连续的字符串，使得最后剩下的字符串只有2020
- 模拟
- 反过来思考一下，无非就是问2020这个串能够通过一次插入若干个字符得到题目所给的串，那么插入的位置要么是在中间要么是在两边。因此，问题我们就可以转化为开头的

几个字符和结尾的几个字符能否拼成2020即可

```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

bool ans[maxn];
const string res = " 2020";
int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int n ; cin >> n;
        string s; cin >> s;
        s = ' ' + s;
        int cnt1 , cnt2;
        cnt1 = cnt2 = 0;
        for ( int i = 1 ; i <= n ; i++ )
        {
            if ( s[i] == res[i] )
                cnt1++;
            else break;
        }
        int d = 4;
        for ( int i = n ; i >= n-3 ; i-- , d-- )
        {
            if ( s[i] == res[d] )
                cnt2++;
            else break;
        }
        if ( cnt1 + cnt2 >= 4 ) ans[cas] = 1;
        else ans[cas] = 0 ;
    }
    for ( int i = 1 ; i <= T ; i++ )
        if ( ans[i] ) cout << "YES" << endl;
        else cout << "NO" << endl;
}
```



```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

LL ans[maxn];

int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int x ; cin >> x;
        if ( x > 45 )
            ans[cas] = -1;
        else if ( x <= 9 )
            ans[cas] = x;
        else
        {
            vector <int> res;
            int d = 9 ;
            while ( x > d )
            {
                res.push_back(d);
                x -= d;
                d --;
            }
            res.push_back(x);
            sort(res.begin(),res.end());
            LL cnt = 0 ;
            for ( auto y : res )
                cnt = cnt * 10 + y;
            ans[cas] = cnt;
        }
    }
    for ( int i = 1 ; i <= T ; i++ )
        cout << ans[i] << endl;
}
```


- 题目中所提供的操作无非就是连续的几个数字求和，要使得最后每个元素都相同，我们把问题转为把连续的几个数字加起来能否组成最后想要的数列当中对应的元素
- 首先我们对整个数列进行求和，同时我们对数列进行求前缀和操作。
- 接着我们找这个数列的和的因数，即有可能最后要的数列的每个元素的大小，通过前缀和进行遍历，看看这种情况是否可行，最后我们取最小的那种方案就行了


```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

int ans[maxn];
int a[maxn];

int pre[maxn];

int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int n ; cin >> n;
        int sum = 0 ;
        memset ( pre , 0 , (n+5)*sizeof(int) );
        for ( int i = 1 ; i <= n ; i++ ) cin >> a[i] , sum += a[i];
        pre[1] = a[1];
        for ( int i = 2 ; i <= n ; i++ ) pre[i] = pre[i-1] + a[i];
        int res = 1 << 30;
        for ( int i = 1 ; i*i <= sum ; i++ )
        {
            if ( sum % i ) continue;
            int fac = sum / i;
            int cur = 0 ;
            bool flag = 1;
            for ( int j = 1 ; j <= n ; j++ )
            {
                if ( pre[j] - pre[cur] > i )
                {
                    flag = 0;
                    break;
                }
                else if ( pre[j] - pre[cur] == i )
                {
                    cur = j ;
                }

                if ( j == n && j != cur )
                    flag = 0 ;
            }
            if ( flag )
                res = min ( res , n -fac );
            cur = 0 ;
            flag = 1;
            for ( int j = 1 ; j <= n ; j++ )
```

```
{
    if ( pre[j] - pre[cur] > fac )
    {
        flag = 0;
        break;
    }
    else if ( pre[j] - pre[cur] == fac )
        cur = j ;

    if ( j == n && j != cur )
        flag = 0 ;
}
if ( flag )
    res = min ( res , n-i );
}
ans[cas] = res;
}
for ( int i = 1 ; i <= T ; i++ )
    cout << ans[i] << endl;
}
```

[E1. Close Tuples (easy version)]


CODE FORCES
 Sponsored by Telegram

DYF3244 | [Logout](#)

[HOME](#) [TOP](#) [CONTESTS](#) [GYM](#) [PROBLEMSET](#) [GROUPS](#) [RATING](#) [EDU](#) [API](#) [CALENDAR](#) [HELP](#) [ABOUT](#) [STATUS](#)

[PROBLEMS](#) [SUBMIT CODE](#) [MY SUBMISSIONS](#) [STATUS](#) [HACKS](#) [STANDINGS](#) [CUSTOM INVOCATION](#)

E1. Close Tuples (easy version)

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the easy version of this problem. The only difference between easy and hard versions is the constraints on k and m (in this version $k = 2$ and $m = 3$). Also, in this version of the problem, you DON'T NEED to output the answer by modulo.

You are given a sequence a of length n consisting of integers from 1 to n . The sequence may contain duplicates (i.e. some elements can be equal).

Find the number of tuples of $m = 3$ elements such that the maximum number in the tuple differs from the minimum by no more than $k = 2$. Formally, you need to find the number of triples of indices $i < j < z$ such that

$$\max(a_i, a_j, a_z) - \min(a_i, a_j, a_z) \leq 2.$$

For example, if $n = 4$ and $a = [1, 2, 4, 3]$, then there are two such triples ($i = 1, j = 2, z = 4$ and $i = 2, j = 3, z = 4$). If $n = 4$ and $a = [1, 1, 1, 1]$, then all four possible triples are suitable.

Input
The first line contains a single integer t ($1 \leq t \leq 2 \cdot 10^5$) — the number of test cases. Then t test cases follow.

The first line of each test case contains an integer n ($1 \leq n \leq 2 \cdot 10^5$) — the length of the sequence a .

The next line contains n integers a_1, a_2, \dots, a_n ($1 \leq a_i \leq n$) — the sequence a .

It is guaranteed that the sum of n for all test cases does not exceed $2 \cdot 10^5$.

Output
Output t answers to the given test cases. Each answer is the required number of triples of elements, such that the maximum value in the triple differs from the minimum by no more than 2. Note that in difference to the hard version of the problem, you don't need to output the answer by modulo. You must output the exact value of the answer.

Example

input

```
4
4
1 2 4 3
4
1 1 1 1
1
1
10
5 6 1 3 2 9 8 1 2 4
```

output

```
2
4
0
15
```

Codeforces Round #690 (Div. 3)

Finished

Practice

→ **Virtual participation**

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

Start virtual contest

→ **Practice**

You are registered for practice. You can solve problems unofficially. Results can be found in the contest status and in the bottom of standings.

→ **Clone Contest to Mashup**

You can clone this contest to a mashup.

Clone Contest

→ **Submit?**

Language: GNU G++17 7.3.0

Choose file: 浏览... 未选择文件.

Submit

→ **Last submissions**

Submission	Time	Verdict
101323794	Dec/15/2020 18:57	Accepted
101323133	Dec/15/2020 18:56	Wrong answer on test 5

→ **Problem tags**

[binary search](#) [combinatorics](#) [math](#)

[sortings](#) [two pointers](#)

No tag edit access

→ **Contest materials**

- Announcement

Codeforces (c) Copyright 2010-2020 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Dec/16/2020 18:08:34 UTC+6 (g1).
Desktop version, switch to [mobile version](#).
[Privacy Policy](#)

Supported by



- 题目大意：给定一个数列，找到其中三个数字，使得三个数字当中最大的数字和最小的数字的差值不会超过2，求这样的三个数字在数列当中有多少个
- 贪心
- 我们对数组进行排序，假设我们得到的一个最小的数字是 a_1 ，最后一个比它大不超过2的位置是 a_s ，也就是说，我们可以从2到s当中挑选两个不一样的数字，那么对答案的贡献就是 C_{s-1}^2

```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

LL ans[maxn];
int a[maxn];
int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int n ; cin >> n;
        for ( int i = 1 ; i <= n ; i++ ) cin >> a[i];
        sort ( a+1 , a+1+n );
        LL res = 0 ;
        for ( int i = 1 ; i <= n-2 ; i++ )
        {
            int pos ;
            if ( a[i] + 2 > a[n] ) pos = n;
            else pos = upper_bound(a+1,a+1+n,a[i]+2) - a - 1;
            LL tmp = pos - i ;
            res += tmp*(tmp-1)/2;
        }
        ans[cas] = res;
    }
    for ( int i = 1 ; i <= T ; i++ )
        cout << ans[i] << endl;
}
```

[E2. Close Tuples (hard version)]


CODE FORCES
 Sponsored by Telegram


 DYF3244 | [Logout](#)

[HOME](#)
[TOP](#)
[CONTESTS](#)
[GYM](#)
[PROBLEMSET](#)
[GROUPS](#)
[RATING](#)
[EDU](#)
[API](#)
[CALENDAR](#)
[HELP](#)
[ABOUT](#)
[STATUS](#)

[PROBLEMS](#)
[SUBMIT CODE](#)
[MY SUBMISSIONS](#)
[STATUS](#)
[HACKS](#)
[STANDINGS](#)
[CUSTOM INVOCATION](#)

E2. Close Tuples (hard version)

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the hard version of this problem. The only difference between the easy and hard versions is the constraints on k and m . In this version of the problem, you need to output the answer by modulo $10^9 + 7$.

You are given a sequence a of length n consisting of integers from 1 to n . The sequence may contain duplicates (i.e. some elements can be equal).

Find the number of tuples of m elements such that the maximum number in the tuple differs from the minimum by no more than k . Formally, you need to find the number of tuples of m indices $i_1 < i_2 < \dots < i_m$, such that

$$\max(a_{i_1}, a_{i_2}, \dots, a_{i_m}) - \min(a_{i_1}, a_{i_2}, \dots, a_{i_m}) \leq k.$$

For example, if $n = 4$, $m = 3$, $k = 2$, $a = [1, 2, 4, 3]$, then there are two such triples ($i = 1, j = 2, z = 4$ and $i = 2, j = 3, z = 4$). If $n = 4$, $m = 2$, $k = 1$, $a = [1, 1, 1, 1]$, then all six possible pairs are suitable.

As the result can be very large, you should print the value modulo $10^9 + 7$ (the remainder when divided by $10^9 + 7$).

Input
The first line contains a single integer t ($1 \leq t \leq 2 \cdot 10^5$) — the number of test cases. Then t test cases follow.

The first line of each test case contains three integers n , m , k ($1 \leq n \leq 2 \cdot 10^5$, $1 \leq m \leq 100$, $1 \leq k \leq n$) — the length of the sequence a , number of elements in the tuples and the maximum difference of elements in the tuple.

The next line contains n integers a_1, a_2, \dots, a_n ($1 \leq a_i \leq n$) — the sequence a .

It is guaranteed that the sum of n for all test cases does not exceed $2 \cdot 10^5$.

Output
Output t answers to the given test cases. Each answer is the required number of tuples of m elements modulo $10^9 + 7$, such that the maximum value in the tuple differs from the minimum by no more than k .

Example

input

```
4
4 3 2
1 2 4 3
4 2 1
1 1 1 1
1 1 1
1
10 4 3
5 6 1 3 2 9 8 1 2 4
```

output

```
2
6
1
20
```

Codeforces Round #690 (Div. 3)

Finished

Practice

★

→ **Virtual participation**

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

[Start virtual contest](#)

→ **Practice**

You are registered for practice. You can solve problems unofficially. Results can be found in the contest status and in the bottom of standings.

→ **Clone Contest to Mashup**

You can clone this contest to a mashup.

[Clone Contest](#)

→ **Submit?**

Language: GNU G++17 7.3.0

Choose file: 浏览... 未选择文件.

[Submit](#)

→ **Last submissions**

Submission	Time	Verdict
101332104	Dec/15/2020 19:13	Accepted

→ **Problem tags**

binary search combinatorics implementation math sortings two pointers

No tag edit access

→ **Contest materials**

- Announcement

Codeforces (c) Copyright 2010-2020 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Dec/16/2020 18:08:46^{UTC+8} (g1).
Desktop version, switch to [mobile version](#).
[Privacy Policy](#)

Supported by




- 接E1
- 进一步地，如果要跳远 m 个数字，我们只要找到了满足条件的 a_s ，我们对答案的贡献就是 C_{s-1}^{m-1}

```

#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0)
#define LL long long
#define maxn (int)(2e5 + 10)

const LL mod = 1e9 + 7;
int m , k ;
LL fac[maxn];
LL inv[maxn];
LL quick_pow ( LL n , LL k )
{
    LL res = 1 ;
    while ( k )
    {
        if ( k & 1 ) res = res * n % mod;
        n = n * n % mod;
        k >>= 1;
    }
    return res;
}

LL get_inv ( LL n ) {
    return quick_pow ( n , mod-2 );
}

void get_fac()
{
    fac[0] = 1;
    for ( int i = 1 ; i <= 2e5 ; i++ )
        fac[i] = fac[i-1] * i % mod;
    inv[200000] = get_inv(fac[200000]);
    for ( int i = 200000-1 ; i >= 0 ; i-- )
        inv[i] = inv[i+1] * (i+1) % mod;
}

LL C ( int n ) {
    if ( n < m-1 ) return 0;
    return fac[n] * inv[m-1] % mod * inv[n-m+1] % mod;
}

LL ans[maxn];
int a[maxn];
int main ()
{
    IOS;
    get_fac();
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {

```

```
int n ; cin >> n;
cin >> m >> k;
for ( int i = 1 ; i <= n ; i++ ) cin >> a[i];
sort ( a+1 , a+1+n );
LL res = 0 ;
for ( int i = 1 ; i <= n-m+1 ; i++ )
{
    int pos ;
    if ( a[i] + k > a[n] ) pos = n;
    else pos = upper_bound(a+1,a+1+n,a[i]+k) - a - 1;
    LL tmp = pos - i ;
    res = ( res + C(tmp) ) % mod;
}
ans[cas] = res;
}
for ( int i = 1 ; i <= T ; i++ )
    cout << ans[i] << endl;
}
```


[F. The Treasure of The Segments]


CODE FORCES
 Sponsored by Telegram

DYF3244 | [Logout](#)

[HOME](#) [TOP](#) [CONTESTS](#) [GYM](#) [PROBLEMSET](#) [GROUPS](#) [RATING](#) [EDU](#) [API](#) [CALENDAR](#) [HELP](#) [ABOUT](#) [STATUS](#)

[PROBLEMS](#) [SUBMIT CODE](#) [MY SUBMISSIONS](#) [STATUS](#) [HACKS](#) [STANDINGS](#) [CUSTOM INVOCATION](#)

F. The Treasure of The Segments

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp found n segments on the street. A segment with the index i is described by two integers l_i and r_i — coordinates of the beginning and end of the segment, respectively. Polycarp realized that he didn't need all the segments, so he wanted to delete some of them.

Polycarp believes that a set of k segments is good if there is a segment $[l_i, r_i]$ ($1 \leq i \leq k$) from the set, such that it intersects every segment from the set (the intersection must be a **point or segment**). For example, a set of 3 segments $[[1, 4], [2, 3], [3, 6]]$ is good, since the segment $[2, 3]$ intersects each segment from the set. Set of 4 segments $[[1, 2], [2, 3], [3, 5], [4, 5]]$ is not good.

Polycarp wonders, what is the minimum number of segments he has to delete so that the remaining segments form a good set?

Input
The first line contains a single integer t ($1 \leq t \leq 2 \cdot 10^5$) — number of test cases. Then t test cases follow.

The first line of each test case contains a single integer n ($1 \leq n \leq 2 \cdot 10^5$) — the number of segments. This is followed by n lines describing the segments.

Each segment is described by two integers l and r ($1 \leq l \leq r \leq 10^9$) — coordinates of the beginning and end of the segment, respectively.

It is guaranteed that the sum of n for all test cases does not exceed $2 \cdot 10^5$.

Output
For each test case, output a single integer — the minimum number of segments that need to be deleted in order for the set of remaining segments to become good.

Example

input

```

4
3
1 4
2 3
3 6
4
1 2
2 3
3 5
4 5
5
1 2
3 8
4 5
6 7
9 10
5
1 5
2 4
3 5
3 8
4 8

```

output

```

0
1
2
0

```

Codeforces Round #690 (Div. 3)

Finished

Practice

→ **Virtual participation**

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

[Start virtual contest](#)

→ **Practice**

You are registered for practice. You can solve problems unofficially. Results can be found in the contest status and in the bottom of standings.

→ **Clone Contest to Mashup**

You can clone this contest to a mashup.

[Clone Contest](#)

→ **Submit?**

Language: GNU G++17 7.3.0

Choose file: [浏览...](#) 未选择文件.

[Submit](#)

→ **Last submissions**

Submission	Time	Verdict
101391735	Dec/16/2020 10:11	Accepted
101386897	Dec/16/2020 09:23	Time limit exceeded on test 8
101386451	Dec/16/2020 09:19	Wrong answer on test 2

→ **Problem tags**

[binary search](#) [data structures](#) [greedy](#)

No tag edit access

→ **Contest materials**

- Announcement

Codeforces (c) Copyright 2010-2020 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: Dec/16/2020 18:08:58^{UTC+8} (g1).
Desktop version, switch to [mobile version](#).
[Privacy Policy](#)

Supported by



- 题目大意：给定若干个线段，求问至少应该删除多少条线段，使得最后留下来的线段至少存在一条线段和其他线段都有交界
- 贪心，二分
- 我们对所有线段的左端点进行排序，对所有线段的右端点进行排序。
- 对于某一条线段，我们需要比较它的左端点比所有线段当中多少右端点要小，也就是说，左半边能够至多和多少线段进行交界；同时我们比较右端点比多少线段的左端点

大，也就是说右半边能和多少线段进行交界。我们取能够交界的最大线段，剩下线段我们删去即可

```
#include <bits/stdc++.h>
using namespace std;
#define IOS ios::sync_with_stdio(0);
#define LL long long
#define maxn (int)(2e5 + 10)

int L[maxn] , R[maxn];
int ans[maxn];
int main ()
{
    IOS;
    int T ; cin >> T;
    for ( int cas = 1 ; cas <= T ; cas++ )
    {
        int n ; cin >> n;
        vector <int> l , r;
        for ( int i = 1 ; i <= n ; i++ )
        {
            cin >> L[i] >> R[i];
            l.push_back(L[i]);
            r.push_back(R[i]);
        }

        sort ( l.begin() , l.end() );
        sort ( r.begin() , r.end() );
        int res = 1 << 30;
        int cnt = 0 ;
        for ( int i = 1 ; i <= n ; i++ )
        {
            cnt = 0 ;
            int pos = lower_bound(r.begin(),r.end(),L[i]) - r.begin();
            cnt += pos + (r[pos] < L[i]);
            pos = upper_bound(l.begin(),l.end(), R[i]) - l.begin();
            cnt += n - pos;
            res = min(cnt, res);
        }
        ans[cas] = res;
        l.clear() , r.clear();
    }
    for ( int i = 1 ; i <= T ; i++ )
        cout << ans[i] << endl;
}
```