Problem A. Queries about less or equal elements

Time limit 2000 ms
Mem limit 262144 kB

Problem Link

You are given two arrays of integers a and b. For each element of the second array b_j you should find the number of elements in array a that are less than or equal to the value b_j.

Input

The first line contains two integers n, m ($1 \le n, m \le 2 \cdot 10^5$) — the sizes of arrays a and b.

The second line contains n integers — the elements of array a ($-10^9 \le a_i \le 10^9$).

The third line contains m integers — the elements of array b ($-10^9 \le b_j \le 10^9$).

Output

Print m integers, separated by spaces: the j-th of which is equal to the number of such elements in array a that are less than or equal to the value b_j.

Examples

Input	Output
5 4 1 3 5 7 9 6 4 2 8	3 2 1 4

Input	Output
5 5 1 2 1 2 5 3 1 4 1 5	4 2 4 2 5

Problem B. Sliding Window Median

Time limit 1000 ms
Mem limit 524288 kB

Problem Link

You are given an array of n integers. Your task is to calculate the median of each window of k elements, from left to right.

The median is the middle element when the elements are sorted. If the number of elements is even, there are two possible medians and we assume that the median is the smaller of them.

Input

The first input line contains two integers n and k: the number of elements and the size of the window.

Then there are n integers $x_1, x_2, \ldots, x_n$: the contents of the array.

Output

Print $n - k + 1$ values: the medians.

Constraints

- $1 \leq k \leq n \leq 2 \cdot 10^5$
- $1 \leq x_i \leq 10^9$

Example

Input	Output
8 3 2 4 3 5 8 1 2 1	3 4 5 5 2 1

Problem C. Inversion Count

Time limit	3599 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

Problem Link

Let A[0...n - 1] be an array of n distinct positive integers. If i < j and A[i] > A[j] then the pair (i, j) is called an inversion of A. Given n and an array A your task is to find the number of inversions of A.

Input

The first line contains t, the number of testcases followed by a blank space. Each of the t tests start with a number n (n <= 200000). Then n + 1 lines follow. In the ith line a number A[i - 1] is given (A[i - 1] <= 10^7). The (n + 1)th line is a blank space.

Output

For every test output one line giving the number of inversions of A.

Example

Input	Output
2 3 3 1 2 5 2 3 8 6 1	2 5

Problem D. Sliding Window Cost

Time limit 1000 ms
Mem limit 524288 kB

Problem Link

You are given an array of n integers. Your task is to calculate for each window of k elements, from left to right, the minimum total cost of making all elements equal.

You can increase or decrease each element with cost x where x is the difference between the new and the original value. The total cost is the sum of such costs.

Input

The first input line contains two integers n and k: the number of elements and the size of the window.

Then there are n integers $x_1, x_2, \ldots, x_n$: the contents of the array.

Output

Output $n - k + 1$ values: the costs.

Constraints

- $1 \leq k \leq n \leq 2 \cdot 10^5$
- $1 \leq x_i \leq 10^9$

Example

Input	Output
8 3 2 4 3 5 8 1 2 1	2 2 5 7 7 1

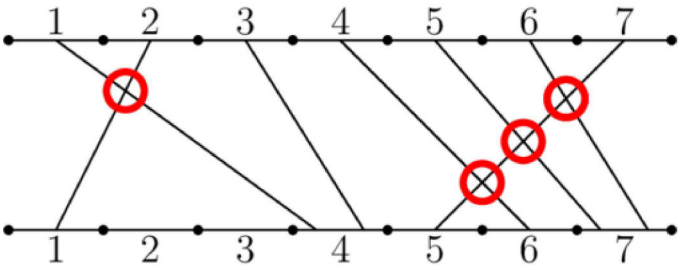
Problem E. Maximum Crossings (Easy Version)

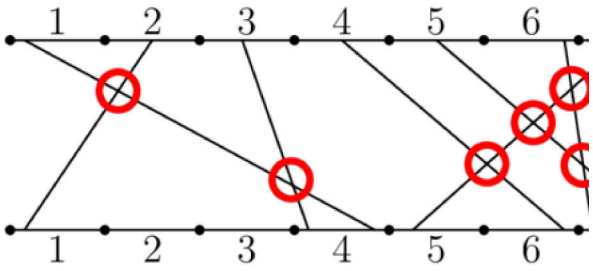
Time limit 1000 ms
Mem limit 262144 kB
Problem Link

The only difference between the two versions is that in this version $n \le 1000$ and the sum of n over all test cases does not exceed 1000.

A *terminal* is a row of n equal segments numbered 1 to n in order. There are two terminals, one above the other.

You are given an array a of length n. For all $i = 1, 2, \ldots, n$, there should be a straight wire from some point on segment i of the top terminal to some point on segment a_i of the bottom terminal. You can't select the endpoints of a segment. For example, the following pictures show two possible wirings if $n = 7$ and $a = [4, 1, 4, 6, 7, 7, 5]$.





A *crossing* occurs when two wires share a point in common. In the picture above, crossings are circled in red.

What is the **maximum** number of crossings there can be if you place the wires optimally?

Input

The first line contains an integer t ($1 \le t \le 1000$) — the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 1000$) — the length of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — the elements of the array.

The sum of n across all test cases does not exceed 1000.

Output

For each test case, output a single integer — the **maximum** number of crossings there can be if you place the wires optimally.

Examples

Input	Output
4 7 4 1 4 6 7 7 5 2 2 1 1 1 3 2 2 2	6 1 0 3

Note

The first test case is shown in the second picture in the statement.

In the second test case, the only wiring possible has the two wires cross, so the answer is 1.

In the third test case, the only wiring possible has one wire, so the answer is 0.

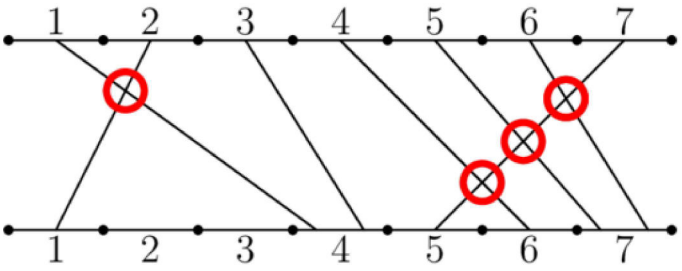
Problem F. Maximum Crossings (Hard Version)

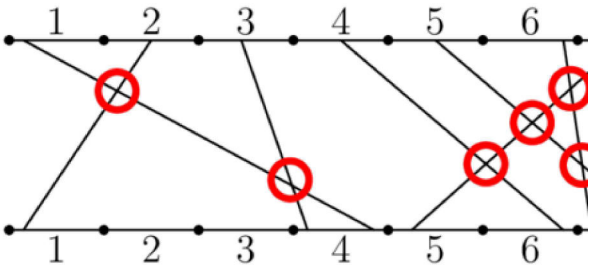
Time limit 1000 ms
Mem limit 262144 kB
Problem Link

The only difference between the two versions is that in this version $n \le 2 \cdot 10^5$ and the sum of n over all test cases does not exceed $2 \cdot 10^5$.

A *terminal* is a row of n equal segments numbered 1 to n in order. There are two terminals, one above the other.

You are given an array a of length n. For all $i = 1, 2, \ldots, n$, there should be a straight wire from some point on segment i of the top terminal to some point on segment a_i of the bottom terminal. You can't select the endpoints of a segment. For example, the following pictures show two possible wirings if $n = 7$ and $a = [4, 1, 4, 6, 7, 7, 5]$.





A *crossing* occurs when two wires share a point in common. In the picture above, crossings are circled in red.

What is the **maximum** number of crossings there can be if you place the wires optimally?

Input

The first line contains an integer t ($1 \le t \le 1000$) — the number of test cases.

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

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — the elements of the array.

The sum of n across all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer — the **maximum** number of crossings there can be if you place the wires optimally.

Examples

Input	Output
4 7 4 1 4 6 7 7 5 2 2 1 1 1 3 2 2 2	6 1 0 3

Note

The first test case is shown in the second picture in the statement.

In the second test case, the only wiring possible has the two wires cross, so the answer is 1.

In the third test case, the only wiring possible has one wire, so the answer is 0.

Problem G. Greetings

Time limit 5000 ms
Mem limit 262144 kB

Problem Link

There are n people on the number line; the i-th person is at point a_i and wants to go to point b_i. For each person, $a_i < b_i$, and the starting and ending points of all people are distinct. (That is, all of the $2n$ numbers $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ are distinct.)

All the people will start moving simultaneously at a speed of 1 unit per second until they reach their final point b_i. When two people meet at the same point, they will greet each other once. How many greetings will there be?

Note that a person can still greet other people even if they have reached their final point.

Input

The first line of the input contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. The description of test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of people.

Then n lines follow, the i-th of which contains two integers a_i and b_i ($-10^9 \le a_i < b_i \le 10^9$) — the starting and ending positions of each person.

For each test case, all of the $2n$ numbers $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ are distinct.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer denoting the number of greetings that will happen.

Examples

Input	Output
5 2 2 3 1 4 6 2 6 3 9 4 5 1 8 7 10 -2 100 4 -10 10 -5 5 -12 12 -13 13 5 -4 9 -2 5 3 4 6 7 8 10 4 1 2 3 4 5 6 7 8	1 9 6 4 0

Note

In the first test case, the two people will meet at point 3 and greet each other.

Problem H. String Reversal

Time limit 2000 ms
Mem limit 262144 kB

Problem Link

You are given a string s. You have to reverse it — that is, the first letter should become equal to the last letter before the reversal, the second letter should become equal to the second-to-last letter before the reversal — and so on. For example, if your goal is to reverse the string "abddea", you should get the string "aeddba". To accomplish your goal, you can swap the **neighboring elements of the string**.

Your task is to calculate the minimum number of swaps you have to perform to reverse the given string.

Input

The first line contains one integer n ($2 \le n \le 200\,000$) — the length of s.

The second line contains s — a string consisting of n lowercase Latin letters.

Output

Print one integer — the minimum number of swaps of neighboring elements you have to perform to reverse the string.

Examples

Input	Output
5 aaaza	2

Input	Output
6 cbaabc	0

Input	Output
9 icpcsguru	30

Note

In the first example, you have to swap the third and the fourth elements, so the string becomes "aazaa". Then you have to swap the second and the third elements, so the string becomes "azaaa". So, it is possible to reverse the string in two swaps.

Since the string in the second example is a palindrome, you don't have to do anything to reverse it.

Problem I. Enemy is weak

Time limit	5000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Problem Link

The Romans have attacked again. This time they are much more than the Persians but Shapur is ready to defeat them. He says: "A lion is never afraid of a hundred sheep".

Nevertheless Shapur has to find weaknesses in the Roman army to defeat them. So he gives the army a weakness number.

In Shapur's opinion the weakness of an army is equal to the number of triplets i, j, k such that $i < j < k$ and $a_i > a_j > a_k$ where a_x is the power of man standing at position x. The Roman army has one special trait — powers of all the people in it are distinct.

Help Shapur find out how weak the Romans are.

Input

The first line of input contains a single number n ($3 \le n \le 10^6$) — the number of men in Roman army. Next line contains n different positive integers a_i ($1 \le i \le n$, $1 \le a_i \le 10^9$) — powers of men in the Roman army.

Output

A single integer number, the weakness of the Roman army.

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cout` (also you may use `%I64d`).

Examples

Input	Output
3 3 2 1	1

Input	Output
3 2 3 1	0

Input	Output
4 10 8 3 1	4

Input	Output
4 1 5 4 3	1