Codeforces Round 970 (Div. 3)

A. Sakurako's Exam

1 second, 256 megabytes

Today, Sakurako has a math exam. The teacher gave the array, consisting of a ones and b twos.

In an array, Sakurako **must** place either a '+' or a '−' in front of each element so that the sum of all elements in the array equals 0.

Sakurako is not sure if it is possible to solve this problem, so determine whether there is a way to assign signs such that the sum of all elements in the array equals 0.

Input

The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases.

The only line of each test case contains two integers a and b ($0 \le a, b < 10$) — the number of '1's and the number of '2's in the array.

Output

For each test case, output "Yes" if you can make the sum of the entire array equal to 0, and "No" otherwise.

You can output each letter in any case (lowercase or uppercase). For example, the strings "yEs", "yes", "Yes", and "YES" will be accepted as a positive answer.

input
5 0 1 0 3 2 0 2 3 3 1

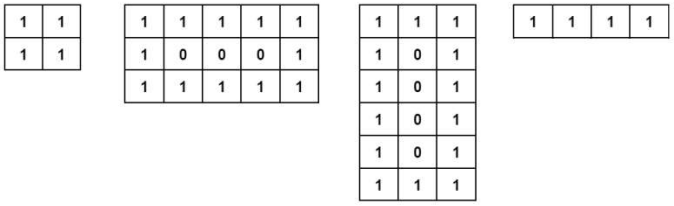
output
NO NO YES YES NO

1. $a = 0, b = 1$: This means the array is $[2]$ — it is impossible to add the signs '+' or '−' to get 0 as a result;
2. $a = 0, b = 3$: This means the array is $[2, 2, 2]$ — it is impossible to add the signs '+' or '−' to get 0 as a result;
3. $a = 2, b = 0$: This means the array is $[1, 1]$ — it is possible to add the signs '+' or '−' to get 0 as a result ($+1 - 1 = 0$);
4. $a = 2, b = 3$: This means the array is $[1, 1, 2, 2, 2]$ — it is possible to add the signs '+' or '−' to get 0 as a result ($+1 + 1 - 2 - 2 + 2 = 0$);

B. Square or Not

2 seconds, 256 megabytes

A beautiful binary matrix is a matrix that has ones on its edges and zeros inside.



Examples of four beautiful binary matrices.

Today, Sakurako was playing with a beautiful binary matrix of size $r \times c$ and created a binary string s by writing down all the rows of the matrix, starting from the first and ending with the r-th. More formally, the element from the matrix in the i-th row and j-th column corresponds to the $((i - 1) * c + j)$-th element of the string.

You need to check whether the beautiful matrix from which the string s was obtained could be **squared**. In other words, you need to check whether the string s could have been build from a **square** beautiful binary matrix (i.e., one where $r = c$).

Input

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

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

The second line of each test case contains the string s of length n. **The string is always the result of writing out the strings of a beautiful matrix.**

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

Output

Print "Yes", if the original matrix could have been square, and "No" otherwise.

input
5 2 11 4 1111 9 111101111 9 111111111 12 111110011111

output
No Yes Yes No No

For the second test case, string 1111 can be obtained from the matrix:

1	1
1	1

For the third test case, string 111101111 can be obtained from the matrix:

1	1	1
1	0	1
1	1	1

There is no square matrix in the fourth case, such that the string can be obtained from it.

C. Longest Good Array

2 seconds, 256 megabytes

Today, Sakurako was studying arrays. An array a of length n is considered good if and only if:

- the array a is increasing, meaning $a_{i-1} < a_i$ for all $2 \le i \le n$;

- the differences between adjacent elements are increasing, meaning $a_i - a_{i-1} < a_{i+1} - a_i$ for all $2 \le i < n$.

Sakurako has come up with boundaries l and r and wants to construct a good array of maximum length, where $l \le a_i \le r$ for all a_i.

Help Sakurako find the maximum length of a good array for the given l and r.

Input

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

The only line of each test case contains two integers l and r ($1 \le l \le r \le 10^9$).

Output

For each test case, output a single integer — the length of the longest good array Sakurako can form given l and r.

input
5 1 2 1 5 2 2 10 20 1 1000000000

output
2 3 1 5 44721

For $l = 1$ and $r = 5$, one possible array could be $(1, 2, 5)$. It can be proven that an array of length 4 does not exist for the given l and r.

For $l = 2$ and $r = 2$, the only possible array is (2).

For $l = 10$ and $r = 20$, the only possible array is $(10, 11, 13, 16, 20)$.

D. Sakurako's Hobby

2 seconds, 256 megabytes

For a certain permutation p* Sakurako calls an integer j reachable from an integer i if it is possible to make i equal to j by assigning $i = p_i$ a certain number of times.

If $p = [3, 5, 6, 1, 2, 4]$, then, for example, 4 is reachable from 1, because: $i = 1 \to i = p_1 = 3 \to i = p_3 = 6 \to i = p_6 = 4$. Now $i = 4$, so 4 is reachable from 1.

Each number in the permutation is colored either black or white.

Sakurako defines the function $F(i)$ as the number of black integers that are reachable from i.

Sakurako is interested in $F(i)$ for each $1 \le i \le n$, but calculating all values becomes very difficult, so she asks you, as her good friend, to compute this.

*A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (the number 2 appears twice in the array), and $[1, 3, 4]$ is also not a permutation ($n = 3$, but the array contains 4).

Input

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

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

The second line of each test case contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$) — the elements of the permutation.

The third line of each test case contains a string s of length n, consisting of '0' and '1'. If $s_i = 0$, then the number p_i is colored black; if $s_i = 1$, then the number p_i is colored white.

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

Output

For each test case, output n integers $F(1), F(2), \ldots, F(n)$.

input
5 1 1 0 5 1 2 4 5 3 10101 5 5 4 1 3 2 10011 6 3 5 6 1 2 4 010000 6 1 2 3 4 5 6 100110

output
1 0 1 1 1 1 2 2 2 2 2 4 1 4 4 1 4 0 1 1 0 0 1

E. Alternating String

2 seconds, 256 megabytes

Sakurako really loves *alternating strings*. She calls a string s of lowercase Latin letters an *alternating string* if characters in the even positions are the same, if characters in the odd positions are the same, and the length of the string is **even**.

For example, the strings 'abab' and 'gg' are *alternating*, while the strings 'aba' and 'ggwp' are not.

As a good friend, you decided to gift such a string, but you couldn't find one. Luckily, you can perform two types of operations on the string:

1. Choose an index i and delete the i-th character from the string, which will reduce the length of the string by 1. This type of operation can be performed **no more than 1 time**;
2. Choose an index i and replace s_i with any other letter.

Since you are in a hurry, you need to determine the minimum number of operations required to make the string an *alternating* one.

Input

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

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

The second line of each test case contains a string s, consisting of lowercase Latin letters.

It is guaranteed that 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 minimum number of operations required to turn the string s into an *alternating* one.

input

```
10
1
a
2
ca
3
aab
5
ababa
6
acdada
9
ejibmyyju
6
bbccbc
6
abacba
5
bcbca
5
dcbdb
```

output

```
1
0
1
1
2
6
2
3
1
1
```

For the string `ababa`, you can delete the first character to get `baba`, which is an *alternating* string.

For the string `acdada`, you can change the first two characters to get `dadada`, which is an *alternating* string.

F. Sakurako's Box

2 seconds, 256 megabytes

Sakurako has a box with n balls. Each ball has it's value. She wants to bet with her friend that if the friend randomly picks two balls from the box (it could be two distinct balls, but they may have the same value), the product of their values will be the same as the number that Sakurako guessed.

Since Sakurako has a PhD in probability, she knows that the best number to pick is the expected value, but she forgot how to calculate it. Help Sakurako and find the expected value of the product of two elements from the array.

It can be shown that the expected value has the form $\frac{P}{Q}$, where P and Q are non-negative integers, and $Q \neq 0$. Report the value of $P \cdot Q^{-1} (\bmod 10^9 + 7)$.

Input

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

The first line of each test case contains a single integer n ($2 \leq n \leq 2 \cdot 10^5$) — the number of elements in the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 10^9$) — the elements of the array.

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

Output

For each test case, output the value of $P \cdot Q^{-1} (\bmod 10^9 + 7)$.

input

```
3
3
3 2 3
4
2 2 2 4
5
1 2 3 4 5
```

output

```
7
6
500000012
```

For the first test, Sakurako's friend can pick these pairs of balls: (a_1, a_2), (a_1, a_3), (a_2, a_3). Their products equal to $3 \cdot 2 = 6$, $3 \cdot 3 = 9$, $3 \cdot 2 = 6$ respectively, so the expected value is $\frac{6+9+6}{3} = 7$.

For the second test, Sakurako's friend can pick these pairs of balls: (a_1, a_2), (a_1, a_3), (a_1, a_4), (a_2, a_3), (a_2, a_4), (a_3, a_4). Their products equal to $2 \cdot 2 = 4$, $2 \cdot 2 = 4$, $2 \cdot 4 = 8$, $2 \cdot 2 = 4$, $2 \cdot 4 = 8$, $2 \cdot 4 = 8$ respectively, so the expected value is $\frac{4+4+8+4+8+8}{6} = \frac{36}{6} = 6$.

G. Sakurako's Task

2 seconds, 256 megabytes

Sakurako has prepared a task for you:

She gives you an array of n integers and allows you to choose i and j such that $i \neq j$ and $a_i \geq a_j$, and then assign $a_i = a_i - a_j$ or $a_i = a_i + a_j$. You can perform this operation any number of times for any i and j, as long as they satisfy the conditions.

Sakurako asks you what is the maximum possible value of mex_k* of the array after any number of operations.

*mex_k is the k-th non-negative integer that is absent in the array. For example, $mex_1(\{1,2,3\}) = 0$, since 0 is the first element that is not in the array, and $mex_2(\{0,2,4\}) = 3$, since 3 is the second element that is not in the array.

Input

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

The first line of each test case contains two integers n and k ($1 \leq n \leq 2 \cdot 10^5, 1 \leq k \leq 10^9$) — the number of elements in the array and the value k for mex_k.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the elements of the array.

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

Output

For each test case, output the maximum mex_k that can be achieved through the operations.

input

```
6
1 3
3
2 10
1 1
3 1
1 2 3
3 2
1 2 4
4 5
2 2 2 16
4 5
2 2 2 3
```

output

```
2
11
3
4
8
8
```

H. Sakurako's Test

1 second, 256 megabytes

Sakurako will soon take a test. The test can be described as an array of integers n and a task on it:

Given an integer x, Sakurako can perform the following operation any number of times:

- Choose an integer i ($1 \le i \le n$) such that $a_i \ge x$;
- Change the value of a_i to $a_i - x$.

Using this operation any number of times, she must find the minimum possible median* of the array a.

Sakurako knows the array but does not know the integer x. Someone let it slip that one of the q values of x will be in the next test, so Sakurako is asking you what the answer is for each such x.

* The median of an array of length n is the element that stands in the middle of the sorted array (at the $\frac{n+2}{2}$-th position for even n, and at the $\frac{n+1}{2}$-th for odd)

Input

The first line contains one integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains two integers n and q ($1 \le n, q \le 10^5$) — the number of elements in the array and the number of queries.

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 following q lines each contain one integer x ($1 \le x \le n$).

It is guaranteed that the sum of n across all test cases does not exceed 10^5. The same guarantee applies to the sum of q across all test cases.

Output

For each test case, output q integers — the answer for each query.

input

```
2
5 5
1 2 3 4 5
1
2
3
4
5
6 3
1 2 6 4 1 3
2
1
5
```

output

```
0 1 1 1 2
1 0 2
```