Codeforces Round 1021 (Div. 2)

A. Vadim's Collection

1 second, 256 megabytes

We call a phone number a *beautiful* if it is a string of 10 digits, where the i-th digit from the left is at least $10 - i$. That is, the first digit must be at least 9, the second at least 8, ..., with the last digit being at least 0.

For example, `9988776655` is a beautiful phone number, while `9099999999` is not, since the second digit, which is 0, is less than 8.

Vadim has a **beautiful** phone number. He wants to rearrange its digits in such a way that the result is the **smallest possible beautiful** phone number. Help Vadim solve this problem.

Please note that the phone numbers are compared as integers.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The only line of each test case contains a single string s of length 10, consisting of digits. It is guaranteed that s is a **beautiful** phone number.

Output

For each test case, output a single string of length 10 — the smallest possible beautiful phone number that Vadim can obtain.

input
4 9999999999 9988776655 9988776650 9899999999

output
9999999999 9876556789 9876567890 9899999999

In the first test case, for the first phone number `9999999999`, regardless of the rearrangement of digits, the same phone number is obtained.

In the second test case, for the phone number `9988776655`, it can be proven that `9876556789` is the smallest phone number that can be obtained by rearranging the digits.

B. Sasha and the Apartment Purchase

1 second, 256 megabytes

Sasha wants to buy an apartment on a street where the houses are numbered from 1 to 10^9 from left to right.

There are n bars on this street, located in houses with numbers $a_1, a_2, \ldots, a_n$. Note that there might be multiple bars in the same house, and in this case, these bars are considered distinct.

Sasha is afraid that by the time he buys the apartment, some bars may close, but **no more than** k bars can close.

For any house with number x, define $f(x)$ as the sum of $|x - y|$ over all open bars y (that is, after closing some bars).

Sasha can potentially buy an apartment in a house with number x (where $1 \le x \le 10^9$) if and only if it is possible to close at most k bars so that after that $f(x)$ becomes minimal among all houses.

Determine how many different houses Sasha can potentially buy an apartment in.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and k ($1 \le n \le 10^5, 0 \le k < n$) — the number of bars and the maximum number of bars that can close.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the house numbers where the bars are located.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output a single integer — the number of houses where Sasha can buy an apartment.

input
4 4 0 1 2 3 4 5 2 7 6 6 7 1 3 1 6 7 9 6 2 5 1 9 10 13 2

output
2 2 4 9

In the first test case, none of the bars can close, so only houses numbered 2 and 3 are suitable. For the house numbered 2, the sum of distances is $|2 - 1| + |2 - 2| + |2 - 3| + |2 - 4| = 4$, and for the house numbered 3, the sum of distances is $|3 - 1| + |3 - 2| + |3 - 3| + |3 - 4| = 4$. However, for the house numbered 1, the sum of distances will be $|1 - 1| + |1 - 2| + |1 - 3| + |1 - 4| = 6$, so the house numbered 1 is not suitable. It can also be proven that Sasha cannot buy apartments in other houses.

In the second test case, the suitable houses are numbered 6 and 7. For Sasha to choose the house numbered 6, it is sufficient that none of the bars close. For Sasha to choose the house numbered 7, the bars in houses 1 and 6 can close. Then the bars will be located in houses numbered 6, 7, and 7.

C. Sports Betting

2 seconds, 256 megabytes

The boarding process for various flights can occur in different ways: either by **bus** or through a **telescopic jet bridge**. Every day, exactly one flight is made from St. Petersburg to Minsk, and Vadim decided to demonstrate to the students that he always knows in advance how the boarding will take place.

Vadim made a bet with n students, and with the i-th student, he made a bet on day a_i. Vadim wins the bet if he correctly predicts the boarding process on both day $a_i + 1$ and day $a_i + 2$.

Although Vadim does not know in advance how the boarding will occur, he really wants to win the bet **at least** with one student and convince him of his predictive abilities. Check if there exists a strategy for Vadim that allows him to **guarantee** success.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the number of students Vadim made bets with.

The second line of each test case contains n integers $a_1, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the days on which Vadim made bets with the students.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, output "Yes" (without quotes) if Vadim can **guarantee** convincing at least one student, and "No" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

input
5 4 1 1 1 1 3 2 2 2 5 2 4 3 2 4 8 6 3 1 1 5 1 2 6 1 1000000000

output
Yes No Yes No No

In the first test case, Vadim needs to make at least one correct prediction about the boarding process on the second and third days. There are a total of 4 possible boarding scenarios for these days, so Vadim can give all 4 students different predictions and guarantee that at least one of them will be correct.

In the second test case, Vadim only made bets with three students and cannot guarantee that he will provide at least one of them with a correct prediction.

D. Baggage Claim

2 seconds, 256 megabytes

Every airport has a baggage claim area, and Balbesovo Airport is no exception. At some point, one of the administrators at Sheremetyevo came up with an unusual idea: to change the traditional shape of the baggage claim conveyor from a carousel to a more complex form.

Suppose that the baggage claim area is represented as a rectangular grid of size $n \times m$. The administration proposed that the path of the conveyor should pass through the cells $p_1, p_2, \ldots, p_{2k+1}$, where $p_i = (x_i, y_i)$.

For each cell p_i and the next cell p_{i+1} (where $1 \le i \le 2k$), these cells must share a common side. Additionally, the path must be simple, meaning that for no pair of indices $i \ne j$ should the cells p_i and p_j coincide.

Unfortunately, the route plan was accidentally spoiled by spilled coffee, and only the cells with odd indices of the path were preserved: $p_1, p_3, p_5, \ldots, p_{2k+1}$. Your task is to determine the number of ways to restore the original complete path $p_1, p_2, \ldots, p_{2k+1}$ given these $k+1$ cells.

Since the answer can be very large, output it modulo $10^9 + 7$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 3 \cdot 10^4$). The description of the test cases follows.

The first line of each test case contains three integers n, m, and k ($1 \le n, m \le 1000$, $n \cdot m \ge 3$, $1 \le k \le \left\lfloor \frac{1}{2}(nm-1) \right\rfloor$) — the dimensions of the grid and a parameter defining the length of the path.

Next, there are $k+1$ lines, the i-th of which contains two integers x_{2i-1} and y_{2i-1} ($1 \le x_{2i-1} \le n$, $1 \le y_{2i-1} \le m$) — the coordinates of the cell p_{2i-1} that lies on the path.

It is guaranteed that all pairs (x_{2i-1}, y_{2i-1}) are distinct.

It is guaranteed that the sum $n \cdot m$ over all test cases does not exceed 10^6.

Output

For each test case, output a single integer — the number of ways to restore the original complete path modulo $10^9 + 7$.

input
5 2 4 2 1 1 2 2 2 4 1 4 1 1 1 1 4 5 5 11 2 5 3 4 4 5 5 4 4 3 5 2 4 1 3 2 2 1 1 2 2 3 1 4 3 4 4 1 2 2 1 3 2 2 3 3 4 3 3 2 2 2 1 1 1 3

output
2 0 2 5 1

In the first test case, there are two possible paths:

- $(1,1) \to (2,1) \to (2,2) \to (2,3) \to (2,4)$
- $(1,1) \to (1,2) \to (2,2) \to (2,3) \to (2,4)$

In the second test case, there is no suitable path, as the cells $(1,1)$ and $(1,4)$ do not have a common neighboring cell.

E. Bermuda Triangle

2 seconds, 256 megabytes

The Bermuda Triangle — a mysterious area in the Atlantic Ocean where, according to rumors, ships and airplanes disappear without a trace. Some blame magnetic anomalies, others — portals to other worlds, but the truth remains hidden in a fog of mysteries.

A regular passenger flight 814 was traveling from Miami to Nassau on a clear sunny day. Nothing foreshadowed trouble until the plane entered a zone of strange flickering fog. Radio communication was interrupted, the instruments spun wildly, and flashes of unearthly light flickered outside the windows.

For simplicity, we will assume that the Bermuda Triangle and the airplane are on a plane, and the vertices of the triangle have coordinates $(0, 0)$, $(0, n)$, and $(n, 0)$. Initially, the airplane is located at the point (x, y) **strictly inside** the Bermuda Triangle and is moving with a velocity vector (v_x, v_y). All instruments have failed, so the crew cannot control the airplane.

The airplane can escape from the triangle if it ever reaches exactly one of the vertices of the triangle. However, if at any moment (possibly non-integer) the airplane hits the boundary of the triangle (but not at a vertex), its velocity vector is immediately reflected relative to that side[†], and the airplane continues to move in the new direction.

Determine whether the airplane can ever escape from the Bermuda Triangle (i.e., reach exactly one of its vertices). If this is possible, also calculate how many times before that moment the airplane will hit the boundary of the triangle (each touch of the boundary, even at the same point, counts; crossing a vertex does not count).

[†] Reflection occurs according to the usual laws of physics. The angle of incidence equals the angle of reflection.

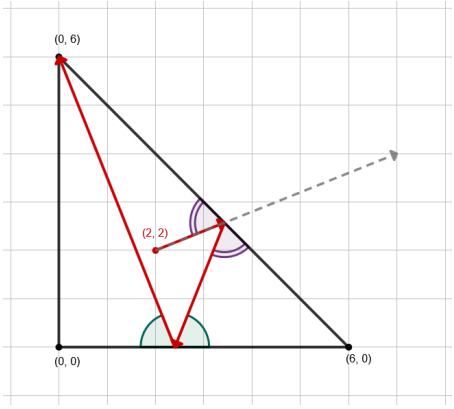
Input
Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The only line of each test case contains five integers n, x, y, v_x, and v_y ($3 \le n \le 10^9$, $1 \le x, y, x + y < n$, $1 \le v_x, v_y \le 10^9$) — numbers describing the vertices of the triangle, the initial coordinates of the airplane, and the initial velocity vector.

Output
For each test case, output a single integer — the number of times the airplane will hit the boundary of the triangle before it escapes. If the airplane never escapes from the triangle, output -1.

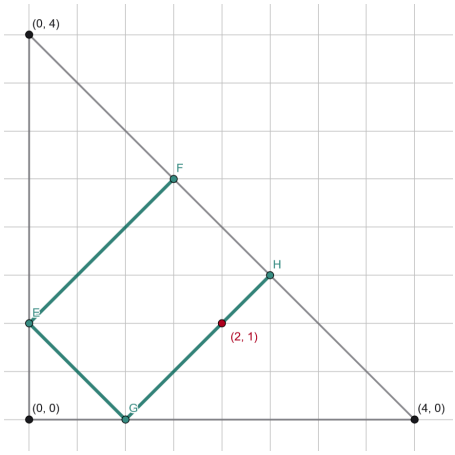
input
6 6 2 2 5 2 6 2 2 20 8 4 1 2 1 1 4 1 1 1 2 4 1 1 2 1 6 2 3 2 3
output
2 2 -1 -1 -1 5

An illustration for the first test case is provided below:



In the second test case, the answer is the same as in the first, as all initial data, except for the speed, are the same, but the airplanes are initially moving in the same direction.

In the third test case, the answer is -1, as the airplane will move exclusively along the segments marked in green. An illustration is provided below:



F. Homework

2 seconds, 256 megabytes

Some teachers work at the educational center "Sirius" while simultaneously studying at the university. In this case, the trip does not exempt them from completing their homework, so they do their homework right on the plane. Artem is one of those teachers, and he was assigned the following homework at the university.

With an arbitrary string a of **even** length m, he can perform the following operation. Artem splits the string a into two halves x and y of equal length, after which he performs **exactly one** of three actions:

- For each $i \in \left\{1, 2, \ldots, \frac{m}{2}\right\}$ assign $x_i = (x_i + y_i) \bmod 2$;
- For each $i \in \left\{1, 2, \ldots, \frac{m}{2}\right\}$ assign $y_i = (x_i + y_i) \bmod 2$;
- Perform an arbitrary number of operations (the same operations defined above, applied recursively) on the strings x and y, independently of each other. Note that in this case, the strings x and y must be of even length.

After that, the string a is replaced by the strings x and y, concatenated in the same order.
Unfortunately, Artem fell asleep on the plane, so you will have to complete his homework. Artem has two binary strings s and t of length n, each consisting of n characters `0` or `1`. Determine whether it is possible to make string s equal to string t with **an arbitrary** number of operations.

Input
Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^5$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^6$) — the length of the strings s and t.

The second line of each test case contains the string s of length n, consisting only of characters `0` and `1`.

The third line of each test case contains the string t of length n, consisting only of characters `0` and `1`.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

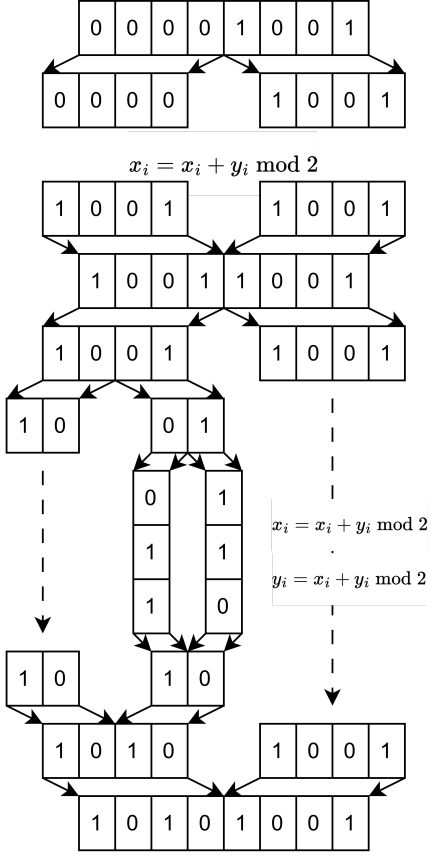
Output
For each test case, output "`Yes`" (without quotes) if it is possible to make string s equal to string t, and "`No`" otherwise.

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
3 8 00001001 10101001 8 00000000 00001001 6 010110 100010

output
Yes No Yes

In the first test case, the string 00001001 can be transformed into the string 10101001 in two operations. The sequence of actions is illustrated in the figure below:



In the second test case, the string 00000000 cannot be transformed into any string other than 00000000, as no non-zero elements can be formed during any operation.