Educational Codeforces Round 84 (Rated for Div. 2)

A. Sum of Odd Integers

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

You are given two integers n and k. Your task is to find if n can be represented as a sum of k **distinct positive odd** (not divisible by 2) integers or not.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \le t \le 10^5$) — the number of test cases.

The next t lines describe test cases. The only line of the test case contains two integers n and k ($1 \le n, k \le 10^7$).

Output

For each test case, print the answer — "YES" (without quotes) if n can be represented as a sum of k **distinct positive odd** (not divisible by 2) integers and "NO" otherwise.

Example

input
6 3 1 4 2 10 3 10 2 16 4 16 5
output
YES YES NO YES YES NO

Note

In the first test case, you can represent 3 as 3.

In the second test case, the only way to represent 4 is $1 + 3$.

In the third test case, you cannot represent 10 as the sum of three distinct positive odd integers.

In the fourth test case, you can represent 10 as $3 + 7$, for example.

In the fifth test case, you can represent 16 as $1 + 3 + 5 + 7$.

In the sixth test case, you cannot represent 16 as the sum of five distinct positive odd integers.

B. Princesses and Princes

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

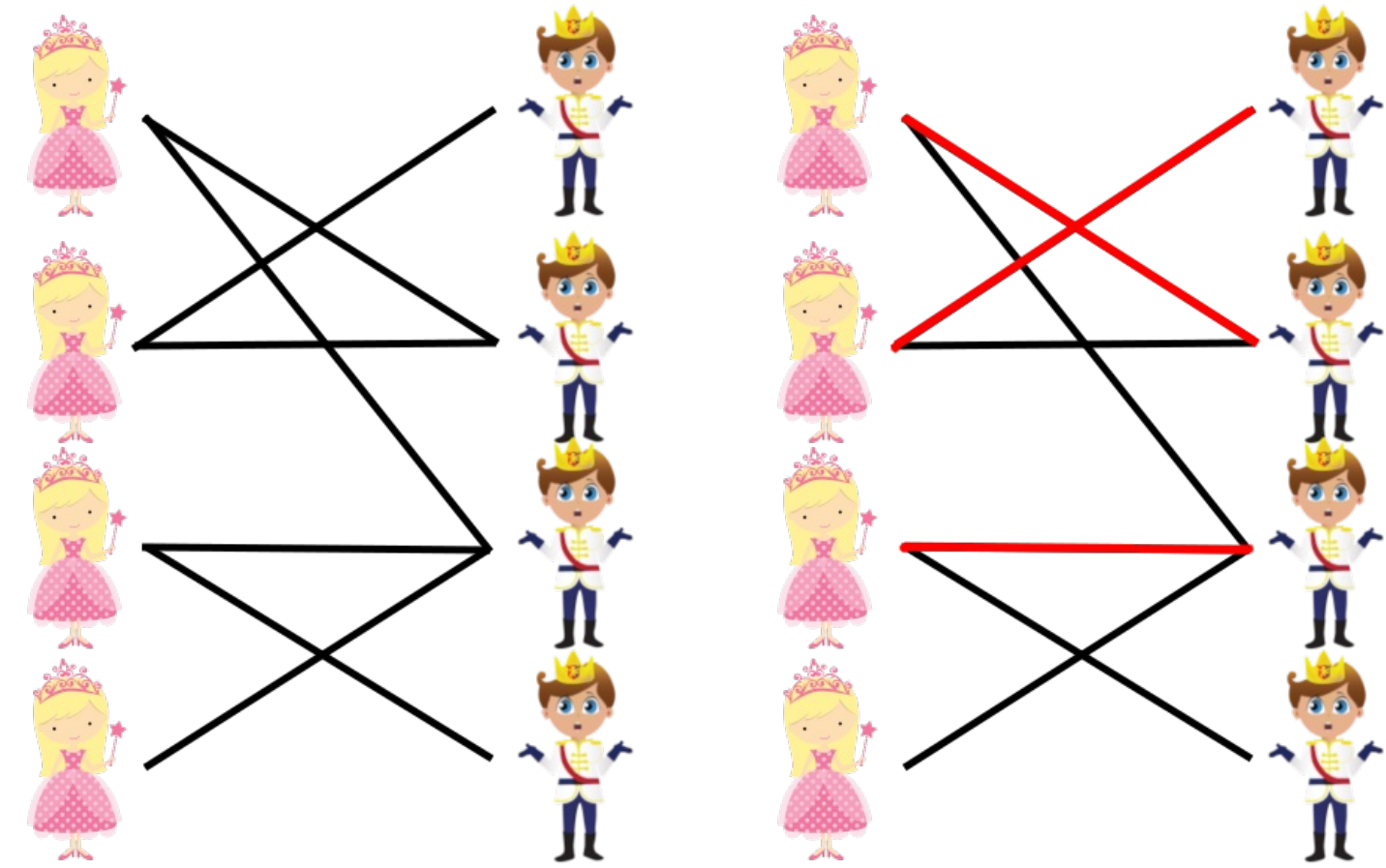
The King of Berland Polycarp LXXXIV has n daughters. To establish his power to the neighbouring kingdoms he wants to marry his daughters to the princes of these kingdoms. As a lucky coincidence there are n other kingdoms as well.

So Polycarp LXXXIV has enumerated his daughters from 1 to n and the kingdoms from 1 to n. For each daughter he has compiled a list of kingdoms princes of which she wanted to marry.

Polycarp LXXXIV is very busy, so he finds a couple for his daughters greedily one after another.

For the first daughter he takes **the kingdom with the lowest number from her list** and marries the daughter to their prince. For the second daughter he takes **the kingdom with the lowest number from her list, prince of which hasn't been taken already**. If there are no free princes in the list then the daughter marries nobody and Polycarp LXXXIV proceeds to the next daughter. The process ends after the n-th daughter.

For example, let there be 4 daughters and kingdoms, the lists daughters have are $[2, 3]$, $[1, 2]$, $[3, 4]$, $[3]$, respectively.



In that case daughter 1 marries the prince of kingdom 2, daughter 2 marries the prince of kingdom 1, daughter 3 marries the prince of kingdom 3, leaving daughter 4 nobody to marry to.

Actually, before starting the marriage process Polycarp LXXXIV has the time to convince one of his daughters that some prince is also worth marrying to. Effectively, that means that he can add exactly one kingdom to exactly one of his daughter's list. **Note that this kingdom should not be present in the daughter's list.**

Polycarp LXXXIV wants to increase the number of married couples.

Unfortunately, what he doesn't have the time for is determining what entry to add. If there is no way to increase the total number of married couples then output that the marriages are already optimal. Otherwise, find such an entry that the total number of married couples increases if Polycarp LXXXIV adds it.

If there are multiple ways to add an entry so that the total number of married couples increases then print any of them.

For your and our convenience you are asked to answer t independent test cases.

Input

The first line contains a single integer t ($1 \le t \le 10^5$) — the number of test cases.

Then t test cases follow.

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

Each of the next n lines contains the description of each daughter's list. The first integer k ($0 \le k \le n$) is the number of entries in the i-th daughter's list. After that k distinct integers follow $g_i[1], g_i[2], \ldots, g_i[k]$ ($1 \le g_i[j] \le n$) — the indices of the kingdoms in the list **in the increasing order** ($g_i[1] < g_i[2] < \cdots < g_i[k]$).

It's guaranteed that the total number of daughters over all test cases does not exceed 10^5.

It's also guaranteed that the total number of kingdoms in lists over all test cases does not exceed 10^5.

Output

For each test case print the answer to it.

Print "IMPROVE" in the first line if Polycarp LXXXIV can add some kingdom to some of his daughter's list so that the total number of married couples increases. The second line then should contain two integers — the index of the daughter and the index of the kingdom Polycarp LXXXIV should add to that daughter's list.

If there are multiple ways to add an entry so that the total number of married couples increases then print any of them.

Otherwise the only line should contain one word "OPTIMAL".

Example

input
5 4 2 2 3 2 1 2 2 3 4 1 3 2 0 0 3 3 1 2 3 3 1 2 3 3 1 2 3 1 1 1 4 1 1 1 2 1 3 1 4

output
IMPROVE 4 4 IMPROVE 1 1 OPTIMAL OPTIMAL OPTIMAL

Note
The first test case is depicted in the statement. Adding the fourth kingdom to the list of the fourth daughter makes her marry the prince of the fourth kingdom.

In the second test case any new entry will increase the number of marriages from 0 to 1.

In the third and the fourth test cases there is no way to add an entry.

In the fifth test case there is no way to change the marriages by adding any entry.

C. Game with Chips

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya has a rectangular Board of size $n \times m$. Initially, k chips are placed on the board, i-th chip is located in the cell at the intersection of sx_i-th row and sy_i-th column.

In one action, Petya can move **all the chips** to the left, right, down or up by 1 cell.

If the chip was in the (x, y) cell, then after the operation:

- left, its coordinates will be $(x, y - 1)$;
- right, its coordinates will be $(x, y + 1)$;
- down, its coordinates will be $(x + 1, y)$;
- up, its coordinates will be $(x - 1, y)$.

If the chip is located by the wall of the board, and the action chosen by Petya moves it towards the wall, then the chip remains in its current position.

Note that several chips can be located in the same cell.

For each chip, Petya chose the position which it should visit. Note that it's not necessary for a chip to end up in this position.

Since Petya does not have a lot of free time, he is ready to do no more than $2nm$ actions.

You have to find out what actions Petya should do so that each chip visits the position that Petya selected for it at least once. Or determine that it is not possible to do this in $2nm$ actions.

Input
The first line contains three integers n, m, k ($1 \le n, m, k \le 200$) — the number of rows and columns of the board and the number of chips, respectively.

The next k lines contains two integers each sx_i, sy_i ($1 \le sx_i \le n, 1 \le sy_i \le m$) — the starting position of the i-th chip.

The next k lines contains two integers each fx_i, fy_i ($1 \le fx_i \le n, 1 \le fy_i \le m$) — the position that the i-chip should visit at least once.

Output

In the first line print the number of operations so that each chip visits the position that Petya selected for it at least once.

In the second line output the sequence of operations. To indicate operations left, right, down, and up, use the characters L, R, D, U respectively.

If the required sequence does not exist, print -1 in the single line.

Examples

input
3 3 2 1 2 2 1 3 3 3 2
output
3 DRD

input
5 4 3 3 4 3 1 3 3 5 3 1 3 1 4
output
9 DDLUUUURR

D. Infinite Path

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

You are given a colored permutation $p_1, p_2, \ldots, p_n$. The i-th element of the permutation has color c_i.

Let's define an *infinite path* as infinite sequence $i, p[i], p[p[i]], p[p[p[i]]] \ldots$ where all elements have **same color** ($c[i] = c[p[i]] = c[p[p[i]]] = \ldots$).

We can also define a multiplication of permutations a and b as permutation $c = a \times b$ where $c[i] = b[a[i]]$. Moreover, we can define a power k of permutation p as $p^k = \underbrace{p \times p \times \cdots \times p}_{k \text{ times}}$.

Find the minimum $k > 0$ such that p^k has at least one infinite path (i.e. there is a position i in p^k such that the sequence starting from i is an infinite path).

It can be proved that the answer always exists.

Input

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

Next $3T$ lines contain test cases — one per three lines. The first line contains single integer n ($1 \le n \le 2 \cdot 10^5$) — the size of the permutation.

The second line contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$, $p_i \ne p_j$ for $i \ne j$) — the permutation p.

The third line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le n$) — the colors of elements of the permutation.

It is guaranteed that the total sum of n doesn't exceed $2 \cdot 10^5$.

Output

Print T integers — one per test case. For each test case print minimum $k > 0$ such that p^k has at least one infinite path.

Example

input
3 4 1 3 4 2 1 2 2 3 5 2 3 4 5 1 1 2 3 4 5 8 7 4 5 6 1 8 3 2

5 3 6 4 7 5 8 4
output
1 5 2

Note

In the first test case, $p^1 = p = [1, 3, 4, 2]$ and the sequence starting from 1: $1, p[1] = 1, \ldots$ is an infinite path.

In the second test case, $p^5 = [1, 2, 3, 4, 5]$ and it obviously contains several infinite paths.

In the third test case, $p^2 = [3, 6, 1, 8, 7, 2, 5, 4]$ and the sequence starting from 4: $4, p^2[4] = 8, p^2[8] = 4, \ldots$ is an infinite path since $c_4 = c_8 = 4$.

E. Count The Blocks

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

You wrote down all integers from 0 to $10^n - 1$, padding them with leading zeroes so their lengths are exactly n. For example, if $n = 3$ then you wrote out `000`, `001`, ..., `998`, `999`.

A block in an integer x is a consecutive segment of equal digits that cannot be extended to the left or to the right.

For example, in the integer 00027734000 there are three blocks of length 1, one block of length 2 and two blocks of length 3.

For all integers i from 1 to n count the number of blocks of length i among the written down integers.

Since these integers may be too large, print them modulo 998244353.

Input

The only line contains one integer n ($1 \le n \le 2 \cdot 10^5$).

Output

In the only line print n integers. The i-th integer is equal to the number of blocks of length i.

Since these integers may be too large, print them modulo 998244353.

Examples

input
1
output
10

input
2
output
180 10

F. AND Segments

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

You are given three integers n, k, m and m conditions $(l_1, r_1, x_1), (l_2, r_2, x_2), \ldots, (l_m, r_m, x_m)$.

Calculate the number of distinct arrays a, consisting of n integers such that:

- $0 \le a_i < 2^k$ for each $1 \le i \le n$;
- bitwise AND of numbers $a[l_i] \& a[l_i + 1] \& \ldots \& a[r_i] = x_i$ for each $1 \le i \le m$.

Two arrays a and b are considered different if there exists such a position i that $a_i \neq b_i$.

The number can be pretty large so print it modulo 998244353.

Input

The first line contains three integers n, k and m ($1 \le n \le 5 \cdot 10^5$, $1 \le k \le 30$, $0 \le m \le 5 \cdot 10^5$) — the length of the array a, the value such that all numbers in a should be smaller than 2^k and the number of conditions, respectively.

Each of the next m lines contains the description of a condition l_i, r_i and x_i ($1 \le l_i \le r_i \le n$, $0 \le x_i < 2^k$) — the borders of the condition segment and the required bitwise AND value on it.

Output
Print a single integer — the number of distinct arrays a that satisfy all the above conditions modulo 998244353.

Examples

input
4 3 2 1 3 3 3 4 6
output
3

input
5 2 3 1 3 2 2 5 0 3 3 3
output
33

Note
You can recall what is a bitwise AND operation here.

In the first example, the answer is the following arrays: $[3, 3, 7, 6]$, $[3, 7, 7, 6]$ and $[7, 3, 7, 6]$.

G. Letters and Question Marks

time limit per test: 4 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

You are given a string S and an array of strings $[t_1, t_2, \ldots, t_k]$. Each string t_i consists of lowercase Latin letters from a to n; S consists of lowercase Latin letters from a to n and **no more than** 14 question marks.

Each string t_i has its cost c_i — an integer number. The value of some string T is calculated as $\sum_{i=1}^{k} F(T, t_i) \cdot c_i$, where $F(T, t_i)$ is the number of occurences of string t_i in T as a substring. For example, $F(\text{aaabaaa}, \text{aa}) = 4$.

You have to replace all question marks in S with **pairwise distinct** lowercase Latin letters from a to n so the value of S is maximum possible.

Input
The first line contains one integer k ($1 \le k \le 1000$) — the number of strings in the array $[t_1, t_2, \ldots, t_k]$.

Then k lines follow, each containing one string t_i (consisting of lowercase Latin letters from a to n) and one integer c_i ($1 \le |t_i| \le 1000$, $-10^6 \le c_i \le 10^6$). The sum of lengths of all strings t_i does not exceed 1000.

The last line contains one string S ($1 \le |S| \le 4 \cdot 10^5$) consisting of lowercase Latin letters from a to n and question marks. The number of question marks in S is not greater than 14.

Output
Print one integer — the maximum value of S after replacing all question marks with **pairwise distinct** lowercase Latin letters from a to n.

Examples

input
4 abc -10 a 1 b 1 c 3 ?b?
output
5

input
2 a 1 a 1 ?

output

2

input

3
a 1
b 3
ab 4
ab

output

8

input

1
a -1
?????????????

output

0

input

1
a -1
??????????????

output

-1