11th KAIST ICPC Mock Competition

Sponsored By:

NAVER D2

STARTLINK

DEVSISTERS

SAMSUNG SOFTWARE MEMBERSHIP

Rules

- This contest is sponsored by Naver D2, Startlink, Devsisters and Samsung Software Membership.
- You can only participate in a team of three students.
- You are allowed to use the Internet and any reference materials.
- You can use more than one computers.
- Problems are NOT sorted by difficulty.
- Solutions to problems submitted for judging are called *runs*. Each run is judged as accepted or rejected, and the team is notified of the results.
- Teams are ranked according to the most problems solved. Teams who solve the same number of problems are ranked first by least total time and, if need be, by the earliest time of submission of the last accepted run.
- The *total time* is the sum of the time *consumed* for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submission of the first accepted run plus 20 penalty minutes for every previously rejected run for that problem. There is no time consumed for a problem that is not solved.
- All problems have the same memory limit, which is language-dependent and is as follows:

```
– C11/C++17      : 1024MB
– Java/Kotlin    : 1536MB
– PyPy 2/Python 3: 2048MB
```

- Problems may have different time limits.

Problem list

#	Problem Name	Time limit (All Languages)
A	Automatic Sprayer 2	2000 ms
B	Dynamic Short Path	12000 ms
C	Equivalent Pipelines	3000 ms
D	Flowerbed Redecoration	3000 ms
E	Goose Coins	3000 ms
F	Histogram Sequence 3	2000 ms
G	Lamb's Respite	3000 ms
H	Or Machine	4000 ms
I	Organizing Beads	2000 ms
J	Periodic Ruler	2000 ms
K	Three Competitions	5000 ms
L	Utilitarianism 2	7000 ms

Problem A. Automatic Sprayer 2

Time limit: 2 seconds

A farm is divided into $n \times n$ unit squares of n rows and n columns. Let's define (i, j) as the unit square in the i-th row and the j-th column ($1 \le i \le n$, $1 \le j \le n$).

The distance between two squares (i_1, j_1) and (i_2, j_2) is defined to be $d\left((i_1, j_1), (i_2, j_2)\right) = |i_1 - i_2| + |j_1 - j_2|$, the Manhattan distance between those two squares.

There are automatic sprayers on this farm that spray fertilizer solution or herbicide so that the owner can produce grain efficiently.

Each sprayer lies entirely in a unit square. The sprayer in (x, y) sprays $A_{x,y}$ liters of solution to all unit squares. $A_{x,y}$ can be any nonnegative integer.

The energy required for the sprayer in (x, y) to spray solution to (i, j) is exactly $d((x, y), (i, j)) \times A_{x,y}$. For each square (i, j), we compute $E_{i,j}$, the sum of energies needed for all sprayers to spray the square (i, j).

Given the matrix E, write a program that generates *any possible* matrix A that corresponds to matrix E. E will be given such that there exists such a matrix A of nonnegative integers whose sum is at most 10^{12}.

Input

The first line contains a single positive integer n ($2 \le n \le 1\,000$).

The next n lines each contain n integers. The j-th ($1 \le j \le n$) integer in the i-th ($1 \le i \le n$) line is $E_{i,j}$ ($0 \le E_{i,j} \le 10^{16}$).

The input is designed such that a matrix A consisting of only non-negative integers whose sum is at most 10^{12} exists which can yield E.

Output

Output n lines, each containing n integers. The y-th ($1 \le y \le n$) integer in the x-th ($1 \le x \le n$) line should be $A_{x,y}$.

Examples

standard input	standard output
5	0 0 0 0 0
4 3 2 3 4	0 0 0 0 0
3 2 1 2 3	0 0 1 0 0
2 1 0 1 2	0 0 0 0 0
3 2 1 2 3	0 0 0 0 0
4 3 2 3 4	
6	0 0 4 0 0 0
43 34 25 24 33 42	0 0 0 0 0 0
42 33 24 23 32 41	0 0 0 0 0 0
41 32 23 22 31 40	0 0 0 0 0 0
40 31 22 21 30 39	0 0 0 5 0 0
39 30 21 20 29 38	0 0 0 0 0 0
48 39 30 29 38 47	

Problem B. Dynamic Short Path

Time limit: 12 seconds

You are given a weighted directed graph with n vertices and $n(n-1)$ edges. For any pair of two distinct vertices $1 \le i, j \le n$, there exists an edge with integer weight $w(i,j)$ where $0 \le w(i,j) \le 2$.

Process the following q queries:

- `1 a b`: Let $dist(a,b)$ be the length of the shortest directed path from vertex a to vertex b $(1 \le a, b \le n)$. Output the value $\min(dist(a,b), 2)$.
- `2 a b c`: Update $w(a,b)$ to c $(1 \le a, b \le n, 0 \le c \le 2, a \ne b)$.

There are at most 2 000 queries of type 2.

Input

The first line contains two integers n and q $(2 \le n \le 600, 1 \le q \le 10^6)$.

The next n lines contain n integers. The j-th integer of the i-th line is $w(i,j)$ $(0 \le w(i,j) \le 2)$. $w(i,i)$ will be denoted as 0 for all i even though no such edge exists.

The next q lines contain several integers denoting the queries in the described form.

There is at least 1 query of type 1.

There are at most 2 000 queries of type 2.

Output

For each query of type 1, output a single integer denoting the answer to that query. Each answer should go on its own line.

Examples

standard input	standard output

5 10
0 1 2 2 2
2 0 2 2 2
2 2 0 1 2
2 2 2 0 2
2 2 2 2 0
1 1 2
1 2 1
1 1 3
1 1 4
1 4 5
1 5 5
2 4 5 0
1 4 5
1 2 5
1 1 5
	1
	2
	2
	2
	2
	0
	0
	2
	2

Problem C. Equivalent Pipelines

Time limit: 3 seconds

You are planning to construct a water pipeline network, connecting n buildings in KAIST. Due to budget problems, you can only use $n-1$ pipes. Each pipe is undirected and connects two different buildings, and all n buildings must be pairwise connected through some sequence of pipes. These pipes form a network.

As a careful planner, you designed d different networks and want to compare them. One can describe each pipe in the network with a durability, which is a single positive integer. Given a network T, define the **vulnerability** $v_T(i,j)$ of two distinct buildings i and j to be the minimum durability of a pipe whose removal separates buildings i and j. In other words, $v_T(i,j)$ is the minimum durability over all pipes on the path connecting i to j.

If two networks T_1 and T_2 satisfy $v_{T_1}(i,j) = v_{T_2}(i,j)$ for all $1 \le i < j \le n$, we say T_1 and T_2 are **equivalent**. To filter out unnecessary plans, group the d designs up to equivalency.

Input

The first line contains two integers d and n ($d \ge 1$, $n \ge 2$, $d \cdot n \le 500\,000$), separated by a space.

From the second line, the descriptions for the d designs are given. Each design is described over $n-1$ lines, each line consisting of three integers a, b and c ($1 \le a, b \le n$, $a \ne b$, $1 \le c \le 10^9$), indicating there is a pipe connecting buildings a and b directly, whose durability is equal to c.

Output

Output d integers in a line. For $1 \le i \le d$, the i-th number should be the minimum index j, where the j-th network in the input is equivalent to the i-th network in the input.

Examples

standard input	standard output

3 3
1 2 1
1 3 1
1 2 1
2 3 1
1 2 1
2 3 2
	1 1 3

3 4
1 2 2
2 3 1
3 4 2
1 3 2
2 4 2
2 3 1
1 2 2
1 3 1
3 4 2
	1 2 1

Problem D. Flowerbed Redecoration

Time limit: 3 seconds

Joon-Pyo decorated a flowerbed in front of his home. The flowerbed is in the shape of an $n \times m$ grid, and one flower is planted in each cell. There are 26 colors, one corresponding to each uppercase letter from A to Z. Suddenly, he wanted to redecorate the flowerbed.

The flowerbed is too large to adjust the flowers one by one. He rented some equipment that can lift and rotate a square plot of land with a side length of d. He planned the construction in the following order, expecting the flowerbed to be properly redecorated.

1. Place the equipment so that exactly the flowers in the first d rows and the first d columns are inside.
2. Rotate the $d \times d$ square inside the equipment 90° clockwise. If this square contains flowers from the last d rows and the last d columns, then the construction is finished. Otherwise, if this square does not contain flowers in the last d columns, move the equipment x squares to the right. Otherwise, move the equipment down by y squares and all the way to the left so it contains flowers from the first d columns.
3. Repeat step 2 until construction is finished.

Note that the equipment will never go out of the flowerbed, as x, y, and d are carefully determined before construction begins.

He cannot start construction without knowing the outcome. Write a program that outputs the result.

Input

On the first line, five integers n, m, y, x, and d are given. ($1 \le n \times m \le 10^6$, $1 \le y \le n$, $1 \le x \le m$, $1 \le d \le \min(n, m)$, $\boldsymbol{n \equiv d \pmod{y}}$, $\boldsymbol{m \equiv d \pmod{x}}$).

Each of the next n lines contains exactly m uppercase letters, the current flowerbed.

Output

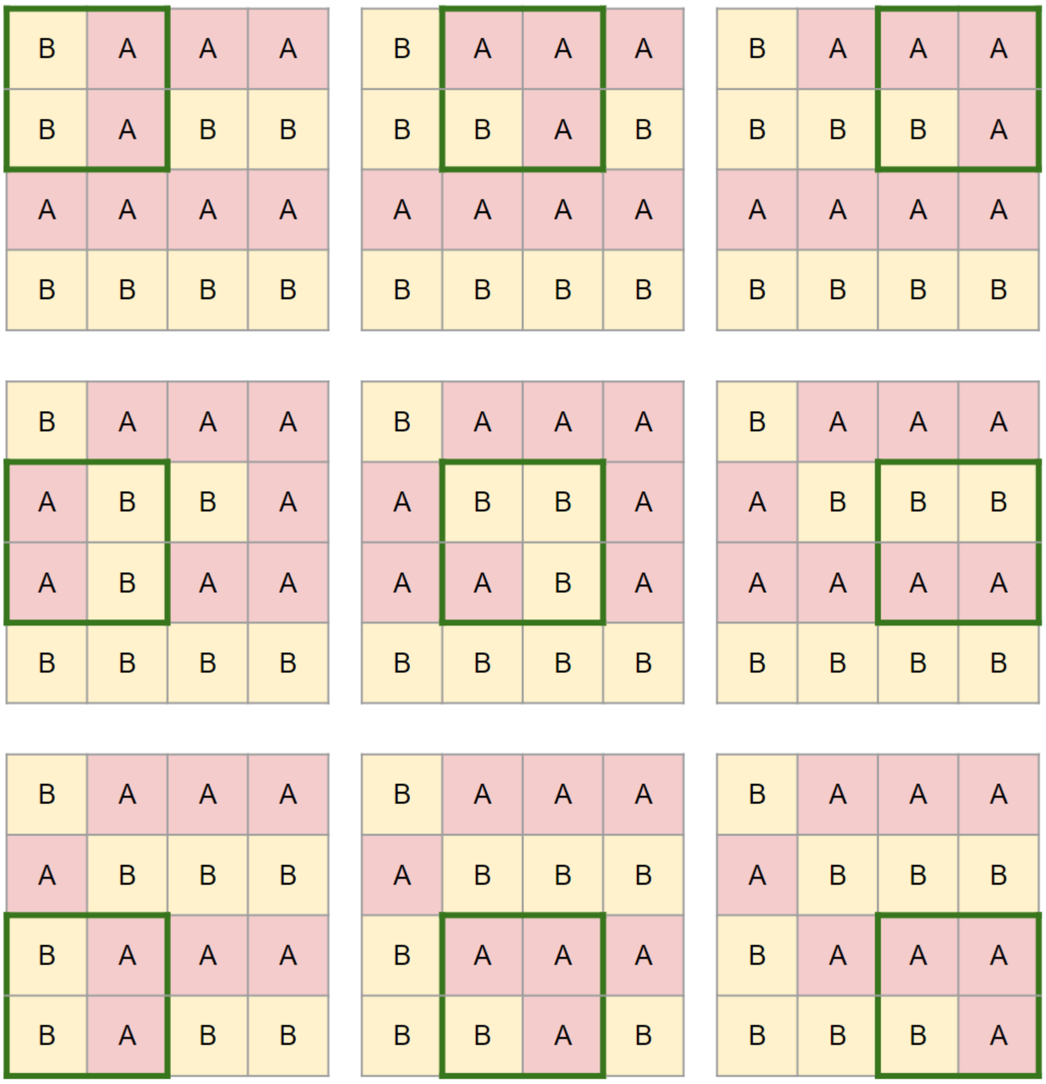
Output n lines, each containing m uppercase letters, the flowerbed after the planned construction.

Examples

standard input	standard output
`4 4 1 1 2`	`BAAA`
`AAAA`	`ABBB`
`BBBB`	`BAAA`
`AAAA`	`BBBA`
`BBBB`	
`6 5 1 2 3`	`PYRBR`
`RBRCY`	`CRCBB`
`YBPBR`	`PPBPY`
`PBRCY`	`CRCYB`
`CYPBR`	`YRBCY`
`PBRCY`	`PYRBR`
`CYPBR`	

Notes

In the first example, the flowerbed changes as follows:



Problem E. Goose Coins

Time limit: 3 seconds

The Goose Kingdom uses n types of goose coins as their national currency. The i-th type of goose coin has a value of c_i goose-dollars and a weight of w_i. For all i $(1 \le i \le n-1)$, c_{i+1} is a multiple of c_i and $c_i < c_{i+1}$.

You visited Goose Market and bought p goose-dollars worth of goods. You want to pay the exact price using exactly k goose coins. You have infinitely many coins of each type, so you don't have to worry about running out of coins.

Write a program to find the minimum and maximum possible total weights of k coins with total value of p goose-dollars. If there is no such set of coins, output -1.

Input

The first line contains three integers n, k, and p $(1 \le n \le 60, 1 \le k \le 10^3, 1 \le p \le 10^{18})$. n is the number of types of goose coins. k is the number of coins you have to use to make exactly p goose-dollars.

In the following n lines, the i-th line contains two integers c_i $(1 \le c_i \le 10^{18})$ and w_i $(1 \le w_i \le 10^{15})$, representing the value and the weight of the i-th type of goose coin.

For all i $(1 \le i \le n-1)$, c_{i+1} is a multiple of c_i and $c_i < c_{i+1}$.

Output

If it is possible to pay exactly p goose-dollars using exactly k goose coins, output the minimum and maximum possible total weights of the k coins. Otherwise, output -1.

Examples

standard input	standard output

3 9 20
1 2
2 5
6 10
	37 44

2 5 10
1 1
3 3
	-1

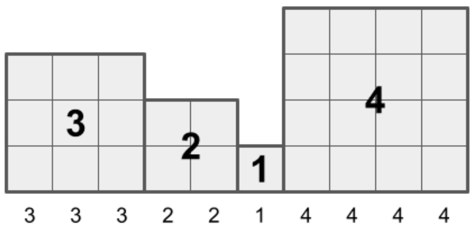
Problem F. Histogram Sequence 3

Time limit: 2 seconds

Consider the histogram composed of n squares with side lengths $a_1, a_2, \cdots, a_n$. Let's call the sequence $(a_1, a_2, \cdots, a_n)$ the *histogram sequence* of this histogram.

Let's consider the height of each column in this histogram. The first a_1 columns will each have height a_1, the following a_2 columns will each have height a_2, ... and the last a_n columns will each have height a_n. Now, let us define the *height sequence* $(b_1, b_2, \cdots, b_{a_1+a_2+\cdots+a_n})$ where b_j $(1 \le j \le a_1 + a_2 + \cdots + a_n)$ is the height of the j-th column.

For example, the histogram with $(3, 2, 1, 4)$ as its histogram sequence has $(3, 3, 3, 2, 2, 1, 4, 4, 4, 4)$ as its height sequence.



Write a program to find the histogram sequence given the height sequence.

Input

The first line contains a single integer m $(1 \le m \le 10^6)$ representing the length of the height sequence $\{b_i\}$ is given.

The second line of the input contains m integers, the height sequence. Specifically, the i-th integer in the line is b_i $(1 \le b_i \le m)$.

The input is designed such that the provided height sequence corresponds to a valid histogram sequence.

Output

Output n integers on a single line, $a_1, a_2, \cdots, a_n$ where $(a_1, a_2, \cdots, a_n)$ is the histogram sequence corresponding to the given height sequence. If there are multiple answers, any one of them will be accepted.

Examples

standard input	standard output
10 3 3 3 2 2 1 4 4 4 4	3 2 1 4
5 2 2 2 2 1	2 2 1

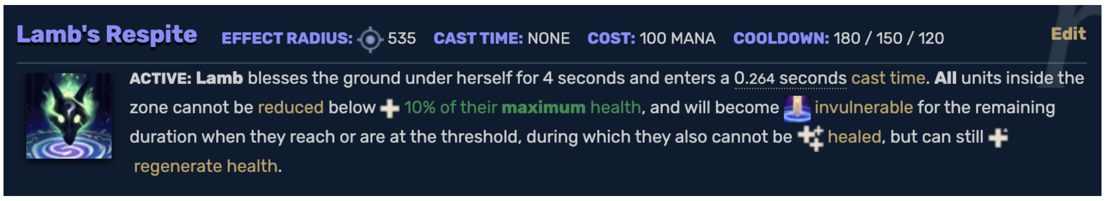
Problem G. Lamb's Respite

Time limit: 3 seconds

Wookje is playing the game *League of Legends*, where two teams of players fight against each other. Each player manages a champion whose status can be represented by two integers: the *health* h, and the *maximum health* H. If a champion's health is less than or equal to zero ($h \le 0$), the champion dies and leaves the game immediately with $h = 0$. If a champion's health is above the maximum health ($h > H$), then it is adjusted to its maximum health. H will always be a positive integer.

In a teamfight, the health of the champion may increase or decrease. More specifically, during a teamfight, the champion was subject to n actions. The i-th ($1 \le i \le n$) action increases the health of the champion by a_i, subject to the rules above. If a_i is positive, it means the champion was healed. If a_i is negative, it means the champion was attacked. If a_i is zero, then nothing happened to the champion.

Wookje's favorite champion in League of Legends is a lamb named *Kindred*. Kindred has a ultimate ability named *Lamb's Respite*. Let's see what this ability does.


Lamb's Respite EFFECT RADIUS: 535 CAST TIME: NONE COST: 100 MANA COOLDOWN: 180 / 150 / 120 Edit

ACTIVE: Lamb blesses the ground under herself for 4 seconds and enters a 0.264 seconds cast time. **All** units inside the zone cannot be reduced below 10% of their **maximum** health, and will become invulnerable for the remaining duration when they reach or are at the threshold, during which they also cannot be healed, but can still regenerate health.

Formally, consider a champion with maximum health H. If Wookje activates *Lamb's Respite* from right before the l-th action until right after the r-th action, a champion's health will never fall below $\lceil \frac{H}{10} \rceil$ during these actions. If a champion's health was less than or equal to $\lceil \frac{H}{10} \rceil$ right before the l-th action, or would hypothetically be less than or equal to $\lceil \frac{H}{10} \rceil$ after the i-th action for some $l \le i \le r$, then its health is set to $\lceil \frac{H}{10} \rceil$ and does not change any further until after the r-th action completes. Otherwise, *Lamb's Respite* does not affect how the champion's health changes.

It is very important to make the right decision on when to activate *Lamb's Respite*. Wookje wants to improve his decision-making skills. However, the teamfights are too complicated, therefore it's hard for him to know when to activate *Lamb's Respite*. To help him, please process the following q queries:

- `1 l r x`: The champion's maximum health is x, and its health starts at x. *Lamb's Respite* is active from right before the l-th action to right after the r-th action. Output the health of the champion after the n actions. If the champion dies, output 0. ($1 \le l \le r \le n, 1 \le x \le 10^9$).
- `2 i x`: Update a_i to x. ($1 \le i \le n, -10^9 \le x \le 10^9$).

Input

The first line contains two integers, n and q ($1 \le n, q \le 300\,000$).

The second line contains n integers. The i-th integer is a_i ($|a_i| \le 10^9$).

The next q lines contain several integers denoting the queries in the described form.

There is at least 1 query of type 1.

Output

For each query of type 1, output a single integer denoting the answer to that query. Each answer should go on its own line.

Examples

standard input	standard output

4 10
0 1 1 -1
1 2 4 2
2 2 -1
1 2 4 2
1 2 3 2
2 1 -1
1 1 4 2
1 2 4 2
2 1 -2
1 1 4 2
1 2 4 2
	1
	1
	0
	1
	1
	1
	0

6 6
2 5 -3 8 1 -4
1 2 5 7
1 1 3 2
2 2 -1
1 4 6 2
2 1 -1
1 1 6 6
	3
	0
	0
	1

Problem H. Or Machine

Time limit: 4 seconds

We are developing the Or Machine, a computer heavily optimized solely for one kind of operation: the `|=` operator in C++'s term.

The Or Machine has n registers, each containing a nonnegative integer less than 2^8. We label them $x_1, x_2, \cdots, x_n$. A program is represented by a list of l operations. Each operation is represented by a pair of integers (a, b), meaning that the machine should update x_a with the bitwise OR of x_a's and x_b's values.

The Or Machine takes a program, the initial values of the registers, and a positive integer t. When run, the program performs each operation in the program one by one. When the last operation is performed, it goes back to the first operation and repeats the process. The machine stops after performing exactly t operations.

We want our machine to be much faster than general-purpose computers, and hardware optimization is probably not enough. Can you help us with some software optimization?

Input

The first line contains three integers, n, l, and t ($1 \le n, l \le 2^{18}$, $1 \le t \le 10^{18}$). l is the length of the program.

The program is given on the next l lines. Each line contains two integers a and b ($1 \le a, b \le n$) representing the pair of registers that participate in the given operation.

The final line contains n integers, the initial values of the registers $x_1, \cdots, x_n$ ($0 \le x_i < 2^8$).

Output

Output n integers on a single line, the values of the registers $x_1, \cdots, x_n$ after t operations.

Examples

standard input	standard output

5 4 5
1 2
2 3
2 4
4 4
8 0 5 3 10
	15 7 5 3 10

Problem I. Organizing Beads

Time limit: 2 seconds

Hyunuk has a long barrel of n ($2 \le n \le 2 \cdot 10^5$) cells. Each cell is either empty or contains a bead. When storing beads, it doesn't look good if they are scattered here and there, so Hyunuk wants to gather all the beads at one end. Specifically, if the barrel has k beads, the beads must be in cells 1 through k or in cells $n - k + 1$ through k.

Hyunuk can move the bead in the i-th cell of the barrel to the $(i-1)$-th cell or the $(i+1)$-th cell by lightly pushing it. If there is a bead in the cell to be moved, that bead is also pushed in the same direction. For example, let there be beads in the 2nd, 3rd, and 5th cells. If Hyunuk pushes the bead in the 2nd cell to the 3rd cell, the bead in the 3rd cell is also pushed to the 4th cell. The bead in the 5th cell remains in place.

Hyunuk wonders how the minimum number of moves required to organize all beads will change when he adds or removes beads from the barrel. Write a program that calculates the minimum number of moves required to organize the bead barrel as Hyunuk inserts or removes beads from the barrel.

Input

The first line contains a single integer n ($2 \le n \le 2 \cdot 10^5$), the length of the bead barrel.

The second line contains a string of length n consisting of only `O`'s and `X`'s representing the state of the bead barrel. If the i-th character of the string is `O`, then the cell has a bead in it. Otherwise, the i-th character of the string is `X` and the cell is empty.

The third line contains a single integer q ($1 \le q \le 2 \cdot 10^5$), the number of actions performed by Hyunuk.

The next q lines each contain a single integer k ($1 \le k \le n$) representing Hyunuk's action. This means that if there is a bead in the k-th cell of the barrel, the bead is removed, and if there is no bead, it is inserted at the corresponding position.

The input will be designed such that the barrel will never have zero beads.

Output

Output q lines, the i-th of which contains a single integer, the minimum number of moves required to organize all beads after Hyunuk's first i actions.

Examples

standard input	standard output
6 OXXOXO 4 3 1 6 3	2 1 2 2

Problem J. Periodic Ruler

Time limit: 2 seconds

Hitagi has a ruler of infinite length. It has a mark on every integer, where the mark on integer i has color c_i. Each color is represented by an integer from 1 to 100.

She noticed that the ruler's color pattern repeats with a period of t. The period t is defined by the **smallest** positive integer that satisfies $c_i = c_{i+t}$ for all integers i.

Hitagi told Koyomi the colors of n marks of her choice. Koyomi wants to find all positive integers that **cannot** be a period of the ruler, regardless of the colors of unchosen marks. Write a program to find all such numbers, and output their count and sum.

Input

The first line contains a single integer n ($1 \le n \le 50$).

The following n lines each contain two integers x_i ($|x_i| \le 10^9$) and a_i ($1 \le a_i \le 100$). This indicates that the integer x_i is marked with the color a_i.

If $i \ne j$, then $x_i \ne x_j$.

Output

Output two integers on one line. The first integer is the number of positive integers that cannot be the period of the ruler. The second integer is their sum.

Examples

standard input	standard output

3
-1 1
1 2
2 1
	2 3

5
1 1
2 1
3 1
4 1
5 1
	4 14

1 1000000000 100	0 0

Problem K. Three Competitions

Time limit: 5 seconds

Last month, n people participated in three competitions. The people are labeled with distinct integers from 1 to n. In each competition, the people were sorted by performance and got ranked from 1 to n. The lower the rank, the better the player. There were no ties in any of the rankings.

Today, instead of n people participating at once, two people compete head-to-head. The winner of the match is the person who wins at least two out of three competitions. The winner then proceeds to compete with another person. This turned out to be quite interesting: even if a person a cannot directly win against another person b, it's possible that another person c wins against b and then a wins against c. That way, we can say that a "indirectly" wins against b. It's also possible that two people can indirectly win against each other!

Formally, a person a is said to **directly win against** another person b if a has a lower rank than b in at least two competitions. Also, a is said to **indirectly win against** b if there exists a sequence of people $p_1, p_2, \cdots, p_k$ ($k \geq 2$) such that p_i directly wins against p_{i+1} for all $i = 1, \cdots, k-1$, $p_1 = a$ and $p_k = b$.

Given the ranks of the people in each competition, answer q questions asking whether person a indirectly wins against another person b.

Input

The first line contains a single integer n ($2 \leq n \leq 2 \cdot 10^5$), the number of people.

Each of the next n lines contains three integers, which represent the ranks of each person in each of the three competitions, in order from person 1 to person n. For each competition, each integer rank from 1 to n appears exactly once.

The next line contains a single integer q ($1 \leq q \leq 2 \cdot 10^5$), the number of questions.

Each of the next q lines contains two integers a and b ($1 \leq a, b \leq n$, $a \neq b$), asking whether person a indirectly wins against person b.

Output

Output q lines. The i-th line should be either `YES` or `NO`. If person a indirectly wins against person b, output `YES`, otherwise output `NO`.

Examples

standard input	standard output

4
2 4 3
3 1 4
4 3 2
1 2 1
3
1 2
2 1
3 4
	YES
	YES
	NO

Notes

Person 1 directly (and indirectly) wins against 2. Person 2 doesn't directly win against 1, but 2 directly wins against 3 and 3 directly wins against 1, so 2 indirectly wins against 1.

Problem L. Utilitarianism 2

Time limit: 7 seconds

To fight the COVID-19 pandemic, vaccines must be transported efficiently to the people. There are n vaccine manufacturers (numbered from 1 to n), m hospitals (numbered from 1 to m), and k agents (numbered from 1 to k) in RUN-land. Agents deliver vaccines from manufacturers to hospitals. The i-th agent has an assignment to deliver c_i vaccines from manufacturer a_i to hospital b_i. The assignments are designed such that for any two agents, they must be assigned to different manufacturers or different hospitals, or both.

RUN-land has a very important law: no manufacturer can use more than one agent, and no hospital can receive vaccines from more than one agent. It is possible that not all agents will be called on to complete their assignments.

The agents play an important role in this battle with the pandemic. Therefore, they should be rewarded reasonably based on their merits. The principle known as *Vickery-Clarke-Groves pricing* states the social value of each agent as follows. Given a set of participating agents S, let $f(S)$ be the maximum possible count of vaccines delivered to hospitals, subject to the law stated above. Let U be the set of all agents. Then the social value of each agent e is $f(U) - f(U \setminus \{e\})$.

In short, given that the agents always act to maximize the number of vaccines that are delivered, the social value of the agent corresponds to the decrease in the number of vaccines delivered if that agent does not participate.

Please determine the social value for all agents.

Input

The first line contains three integers n, m, and k ($1 \le n, m \le 2\,000$, $1 \le k \le \min(8\,000, n \times m)$).

The next k lines contains three integers a_i, b_i, and c_i ($1 \le a_i \le n$, $1 \le b_i \le m$, $1 \le c_i \le 10^{12}$).

If $i \ne j$, $(a_i, b_i) \ne (a_j, b_j)$.

Output

Output k lines, where the i-th line contains a single integer denoting the social value of agent i.

Examples

standard input	standard output

5 5 7
1 4 3
5 1 7
2 2 8
3 5 2
4 3 8
2 3 6
5 4 9
	1
	1
	8
	2
	8
	0
	0