Problem A. Prevent a Galactic War!

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

A long time ago in a galaxy far,
far away....

The Trade Federation is an extremely powerful organization uniting lots of star systems and galaxies under its flag. Among other things there are N branches of industry under the Federation control. Each of them produces a unique kind of product.

During the last year the i-th branch produced x_i tons of its product. Some part of the product was sent to other branches' production, while another part was sold to retailers. It's known that during the previous year the j-th branch used c_{ij} tons of product produced by the i-th branch. Additionally, the i-th branch sold y_i tons of its product to retailers. The Trade Federation controls the process of production very carefully and because of this, there is no shortage nor excess of any product.

The intelligence of the Alliance reported that the Federation is planning to sell $\breve{y}_i$ tons of product from the i-th branch this year. Since profit from the sold products can help the Federation to build a droid army, the Alliance wants to sabotage the production.

You are young Jedi Obi-Wan Kenobi, and you have just received instructions to compute how much product should i-th branch produce including the product sent to other branches. It's known that the amount of other branches' products needed by the i-th branch is directly proportional to the total amount of product produced by the i-th branch. Write a program that, given the values c_{ij}, last year sales y_i and planned sales $\breve{y}_i$, computes how many tons of product $\breve{x}_i$ should the i-th branch produce this year.

May the Force be with you!

Input

The first line of input contains integer N, the number of branches ($1 \le N \le 3000$).

The next N lines contain N integers each, the i-th of them contains integers $c_{i1}, c_{i2}, \ldots, c_{iN}$ ($1 \le c_{ij} \le 1000$); c_{ij} means how many tons of product produced by the i-th branch was used by the j-th branch last year.

The next line contains N integers $y_1, y_2, \ldots, y_N$, the amount of product of each branch that was sold to retailers last year ($10^7 \le y_i \le 10^9$).

The last line of input contains N integers $\breve{y}_1, \breve{y}_2, \ldots, \breve{y}_N$, the amount of product of each branch that is planned to be sold to retailers this year ($10^7 \le \breve{y}_i \le 10^9$).

Output

Print N numbers $\breve{x}_1, \breve{x}_1, \ldots, \breve{x}_N$: the amount of product each branch should produce this year.

The answer is considered correct if $\sqrt{\sum_{i=1}^{n} (\bar{y}_i - \breve{y}_i)^2} < 10^{-4}$, where $\bar{y}_i$ is the amount of product of i-th branch that could be sold if each branch i produces $\breve{x}_i$ tons of product.

Example

standard input
3 1 10 1 11 1 1 1 12 1 10000000 20000000 20000000 30000000 20000000 30000000
standard output
30000014.50000887 20000035.49997725 30000016.50001112

Problem B. Forcefield

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

While Luke Skywalker was trying to acquire confindental data from the imperial flagship, he got into a trap. Now he is standing on the very edge of a long pipe inside the spaceship's reactor. To deal with the situation, he can move only forward. There is a stormtrooper standing at some point in front of Luke, and he have just made a shot from a blaster towards Luke. At the same moment, R2-D2 has accidentally activated the spaceship's forcefield. The forcefield generators are located on the pipe. The generators are powerful enough to change the bolt movement direction.

Each generator reflects a bolt of plasma when it comes from the front side and lets it fly through when it comes from the back. However, if the bolt comes from the front, it destroys the generator. Therefore, there are two types of generators: generators of one type face towards Luke, and generators of the other type face in the opposite direction.

After a generator is destroyed, it no longer affects the bolt. The stormtrooper's batteries are low on charge, so he cannot shoot anymore and now is lying on the bottom of the pipe so that the bolt will not hit him.

When the bolt reaches Luke, he must reflect it with the use of his lightsaber. But because of the forcefield, after being reflected, the bolt just changes direction to the opposite and continues to fly along the pipe. Luke cannot move until all the generators are destroyed. Additionally, if after destroying all the generators, the bolt flies towards Luke again, he must reflect it one more time. So, now he wonders if it's possible to destroy all the generators, and if so, how many times he will have to reflect the plasma bolt.

Input

The first line of the input contains two integers N and X, the number of generators and the distance between Luke and the stormtrooper ($1 \le N \le 100\,000$, $1 \le X \le 10^9$).

The next N lines contain two integers each. The i-th line contains x_i, the distance between Luke and the i-th generator, and the type of the generator ($1 \le x_i \le 10^9$; the type is 1 if the generator is faced towards Luke, or 0 otherwise).

The generators are given in the order of increasing distance from Luke (that is, $x_i < x_j$ if $i < j$). The position of the stormtrooper doesn't coincide with any of the generators.

Output

Print -1 if it's not possible to destroy all the generators.

Otherwise, print a single integer: the number of times Luke will have to reflect the bolt of plasma.

Example

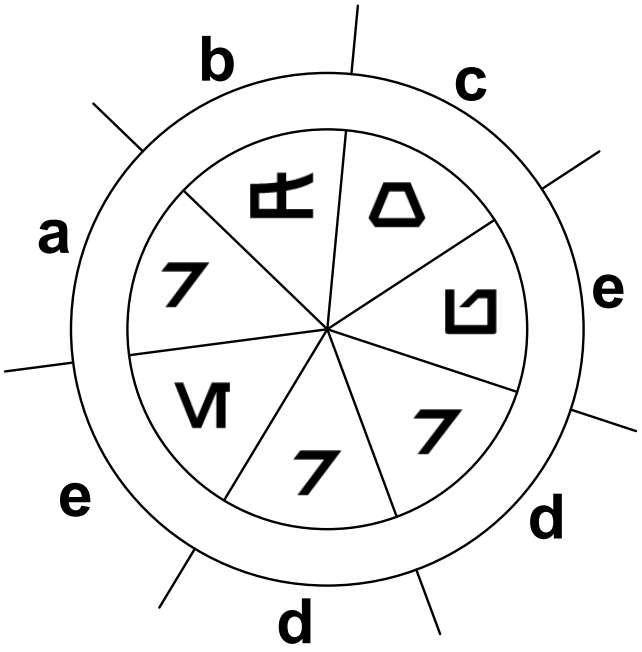
standard input	standard output
2 3 1 1 2 1	3

Problem C. Missing Part

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Recently, Han Solo has found the last missing part of the map of the Galaxy. This part as the shape of a thin disc. The hole in the map where the part should be inserted has the same shape. The problem is that Han Solo doesn't know the angle by which he should rotate this part.

However, there are some marks on the edge of the last part. The circle is divided into N arcs of equal length, and each of them is marked with one of the five hieroglyphs. Han Solo assumed that these marks mean five local space types. Based on this, he looked carefully at the main part of the map, split the circular border of the hole into N arcs of equal length and wrote down the local space type of each arc.



Now Han Solo should match each hieroglyph to one of the types, different hieroglyphs to different types, and insert the last part into the map, so that each arc on the last part will coincide with exactly one arc on the main part. Let the *suspiciousness* of these operations be the number of arcs on which the local space type on the main part differs from the type associated with the corresponding hieroglyph. Han wants to make the suspiciousness as little as possible. Help him to calculate the minimal suspiciousness among all possible placements and matchings.

Note that Han can't flip any part of the map, only rotation is possible.

Input

The first line of input contains one string denoting the hieroglyphs on the last part in clockwise order. For your convenience, the hieroglyphs are replaced with uppercase English letters from "A" to "E".

The second line of input contains one string denoting the local space types that Han Solo assigned to arcs on the main part of the map, also in clockwise order. The types are given as lowercase English letters from "a" to "e".

The lengths of both strings are equal to some integer N ($1 \le N \le 50\,000$).

Output

Print a single integer: the minimal suspiciousness among all possible placements and matchings.

Examples

standard input	standard output
ABCD cdab	0
DABCCEC abcedde	1
ACBDCBABACD babcdbadcab	3

Note

In the first example, the disk should be rotated clockwise by two positions, and the hieroglyphs should be matched in the following way: A → a, B → b, C → c, D → d, E → e. After that, the strings will become equal, and the suspiciousness will be zero.

In the second example, the disk shouldn't be rotated at all, and the hieroglyphs should be matched in the following way: A → b, B → c, C → e, D → a, E → d. After that, the marks on the disk will form the string "abceede". The suspiciousness of such placement and matching is equal to one.

The picture in the statement corresponds to the second example and shows the disk rotated clockwise by one position.

Problem D. Handling a Spaceship

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

This is an interactive problem.

Wedge Antilles has infiltrated a new secret imperial spaceship which uses N-dimensional hyperspace to move around the Galaxy faster than any other spaceship. Fortunately, there is nobody onboard now, but the spaceship is heading fast in some direction!

The spaceship uses N selectors to choose speed, each of them can be set in one of the M positions. The i-th selector corresponds to i-th *direction* X_i, where X_i is a N-dimensional vector; when it is set to j-th position, the vector $K_{ij} \cdot X_i$ is added to the total speed of the spaceship. Here, each K_{ij} is an integer known as *transmission coefficient* of j-th gear on i-th selector. Therefore, the total speed of the spaceship is an N-dimensional vector defined as $S = \sum_{1}^{N} K_{ig_i} X_i$, where g_i is the position the i-th selector is set to.

Wedge knows that, in order to make spaceship driving convenient, the *directions* and *transmission coefficients* of any spaceship, including this one, satisfy some additional constraints. First of all, the *directions* are chosen in such a way that for any desired N-dimensional speed vector S, it's possible to choose *coefficients* c_i such that $S = \sum_{1}^{N} c_i X_i$ (note that c_i may or may not be equal to any of the K_{ij}). Additionally, K_{ij} is always less than K_{it} if $j < t$. And the last but not least, for each selector, one of the *transmission coefficients* is equal to zero.

As we've already mentioned, the spaceship is now moving in some direction through the hyperspace, and Wedge doesn't know the *directions* and the *transmission coefficients*. He wants to stop the spaceship. In order to do that, he needs to find the positions for each selector such that the total speed S will be equal to zero. All he can do now is to set each selector to some position (that is, choose g_i for each i) and compute the resulting speed.

Your program has to play Wedge's role and choose positions of each selector. The jury's program will tell your program the resulting speed. Wedge is limited in time, so you could make no more than 120 queries.

Interaction Protocol

At first, your program will be given two integers N and M ($1 \le N, M \le 100$). After that, your program has to determine positions of each selector that will produce zero speed.

Though in real world the spaceship is very fast, its hyperspace speed is very limited. Wedge knows that all *transmission coefficients* and components of *directions* don't exceed 1000 by absolute value.

To make a query, the program must print the question mark ("`?`"), a space, and then N integers $g_1, g_2, \ldots, g_N$ separated by spaces. Each number must be in the range from 1 to M. This will mean that Wedge sets the first selector to position g_1, the second selector to position g_2, and so on. Then your program must read N integers: the coordinates of the resulting speed vector S.

When your program is ready to print the answer, it must print the English letter "`A`", a space, and then N integers $g_1, g_2, \ldots, g_N$ separated by spaces. After that, your program must exit immediately.

Don't forget to flush the output after each query.

Please note: if for some query, your program receives the number 987 654 321 as the first integer of the answer, it means that your solution is wrong, and your program must exit immediately. In that case, you can hope to receive the proper outcome ("`Wrong Answer`", "`Presentation Error`", etc.). If your program does not exit in such situation, you will most likely receive a random outcome instead of what actually happened.

Examples

standard input	standard output
2 2 0 -1 2 0 0 0	 ? 1 1 ? 2 2 ? 1 2 A 1 2
2 3 0 -2 200 0 0 -1	 ? 1 1 ? 3 3 ? 1 2 A 1 3

Note

In the first example, there are two selectors and two positions for each of them. The *directions* are the following: $X_1 = (1, 0)$, $X_2 = (0, 1)$. The *transmission coefficients* for the first selector are $K_{11} = 0$, $K_{12} = 2$, and for the second selector, they are $K_{21} = -1$, $K_{22} = 0$. Therefore, to achieve zero speed, Wedge needs to set the first selector in the first position, and the second selector in the second position.

In the second example, there are also two selectors, but now, each of them has three different positions. The *directions* and *transmission coefficients* are the following:

$$X_1 = (1, 0), \quad X_2 = (0, 1);$$

$$K_{11} = 0, \quad K_{12} = 100, \quad K_{13} = 200;$$

$$K_{21} = -2, \quad K_{22} = -1, \quad K_{23} = 0.$$

Problem E. Cryptographic Argument

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

To enter the spaceship security system, the astromechanical droid R2-D2 uses a secret cryptographic algorithm, one step of which solves the following problem:

Consider a thin paper strip with $n = 2^k$ cells located in one row. The cells are numbered 0 through $n-1$. Fold the strip in half, the right half under the left one, so that the cell numbered $2^{k-1} - j$ will be above the cell numbered $2^{k-1} + j - 1$ for all j from 1 to 2^{k-1}. Then fold the resulting strip, which has how length 2^{k-1}, again and again in the same way until the length of the strip is one. In the resulting folded strip, the cells are placed above each other in some order. Let the sequence $a_0, a_1, \ldots, a_{n-1}$ be the numbers of the cells from top to bottom.

When the droid connects to the spaceship system, it is given an integer n, and after that, it is asked several queries to get verified. Each query is a segment $[l, r]$ $(0 \le l \le r < n)$ for which the droid must compute the following expression, in which operations + and $\oplus$ alternate:

$$F(l, r) = a_l + a_{l+1} \oplus a_{l+2} + a_{l+3} \oplus a_{l+4} + a_{l+5} \oplus \ldots a_r.$$

Additional care must be taken since operation + has higher priority than $\oplus$ if l is even, and operation $\oplus$ has higher priority than + otherwise. If the droid doesn't answer correctly these queries in one second, he is revealed.

R2-D2 argued with C-3PO that the latter couldn't solve this problem. C-3PO is a protocol droid, it knows six million forms of communication, but nothing about cryptography. Help him to solve the problem in order to surprise R2-D2.

In order to emulate a security system, R2-D2 suggested an algorithm for choosing the segments $[l, r]$. He also suggested to check only the results of hashing $F(l, r)$, but not the values $F(l, r)$ themselves. Let the queries be numbered 0 through $m - 1$. Then these values are computed as follows:

- $h_{j+1} = ((l_j \oplus r_j \oplus h_j \oplus F(l_j, r_j)) + c) \bmod 1\,000\,000\,007$;
- $l_{j+1} = ((l_j \oplus a \oplus h_{j+1}) \bmod (n + 1)) \bmod n$;
- $r_{j+1} = ((r_j \oplus b \oplus h_{j+1}) \bmod (n - l_{j+1})) + l_{j+1}$.

Here, h is the value of the hash function, and $h_0 = 0$. Your task is to compute the final hash value h_m.

Input

The first line of input contains an integer k $(0 \le k \le 30)$.

The second line contains three integers m, l_0, r_0: the number of queries and the bounds of the initial query $(1 \le m \le 10^7,\ 0 \le l_0 \le r_0 < n)$.

The third line contains three integers a, b and c: the parameters of the generating algorithm $(0 \le a, b, c < 2^{30})$.

Output

Print one integer h_m: the value of the hash function after processing all queries.

Examples

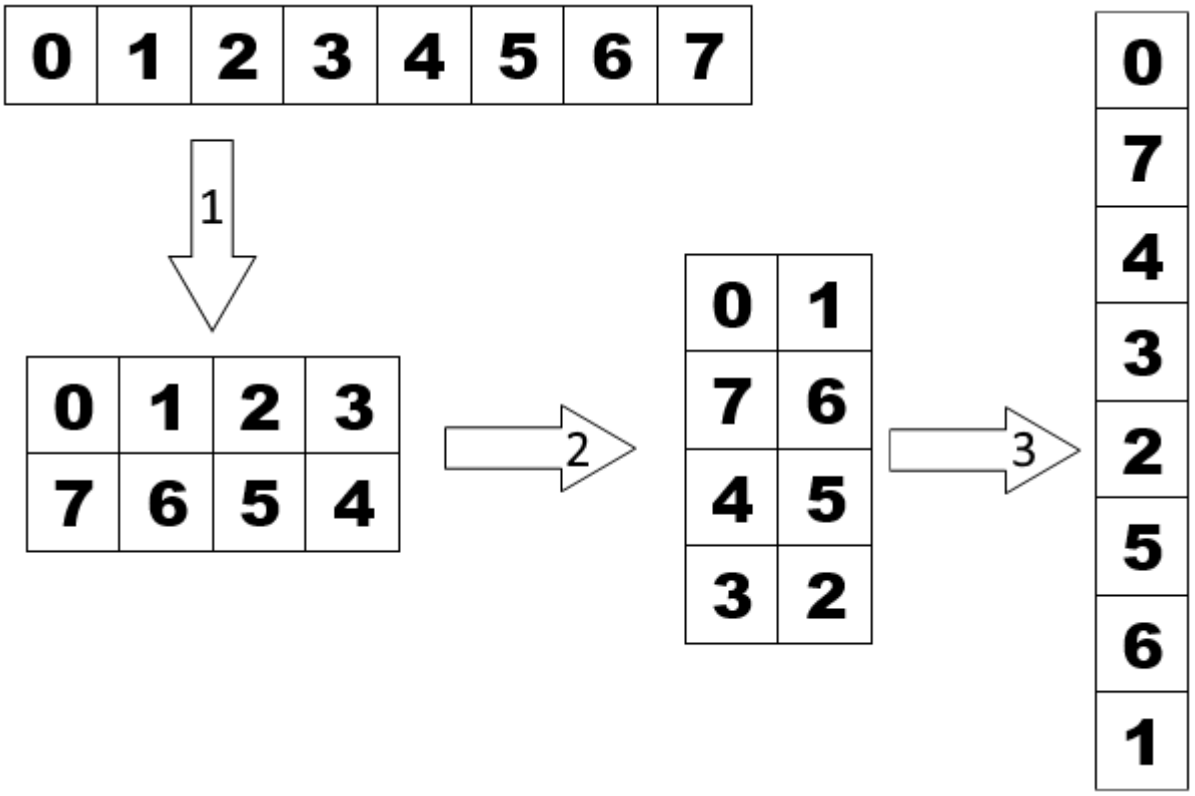
standard input	standard output
3 1 2 6 3 4 5	7
3 709193 4 5 273035200 65685838 991992535	156951996

Note

Operation $\oplus$ means exclusive or.

The results of operations with higher priority must be computed earlier. Operations with the same priority must be processed in the order from left to right.

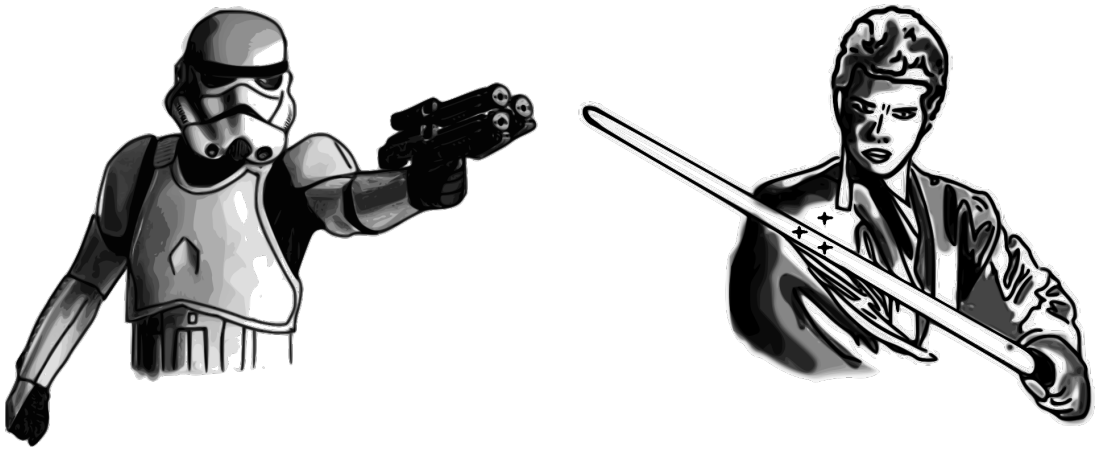
A strip of length $2^3 = 8$ folds like in the picture.



Problem F. The Jedi Killer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Everyone knows a Jedi can reflect blaster bolts at any speed, so a unique anti-Jedi three-blaster was invented. It can produce three shots at once, so it is impossible for a Jedi to reflect them all.



However, a new lightsaber was constructed by adding guards to the usual lightsaber. Now the guards can help in resisting the anti-Jedi blaster, but the blaster can change the location of muzzles. Everyone is now puzzled with the question: how to understand whether a particular lightsaber could reflect all the three bolts from a particular three-blaster or not. Write a program which can answer such questions.

All bolts from three-blaster fly along traces which are straight lines, all three traces are parallel to each other. Consider a plane which is perpendicular to the traces. A lightsaber can be represented as three closed line segments on the plane, one for the main ray with length L_m and two for the guards with lengths L_g, all three segments start from the same point, and the guards' segments are perpendicular to the main segment. You are given lengths L_m and L_g, and also three points on a plane describing the places where traces intersect with the plane. Find if the lightsaber can be placed to cover all the three points or not.

Input

The first line of the input contains T, the number of test cases ($1 \le T \le 10\,000$).

Each test case is given on four lines. Additionally, there is an empty line before each test case.

The first line of each test case contains two integers L_m and L_g ($1 \le L_m \le 30\,000$, $0 \le L_g \le 30\,000$). Each of the following three lines contains two integers; these are the coordinates of the three distinct points.

Each coordinate in the input doesn't exceed 10^4 by its absolute value.

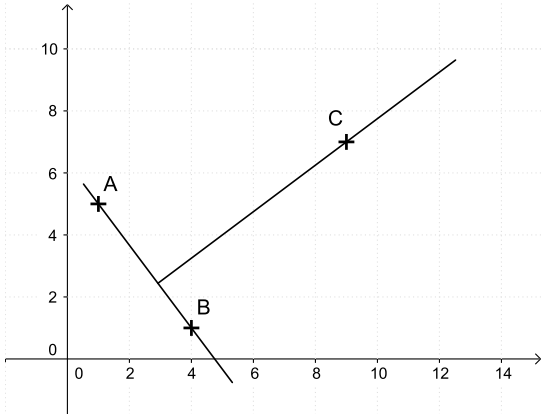
Output

For each test case, print a single line containing "`YES`" (without quotes) if the lightsaber can be placed in such a way that it will reflect all three bolts, and "`NO`" otherwise.

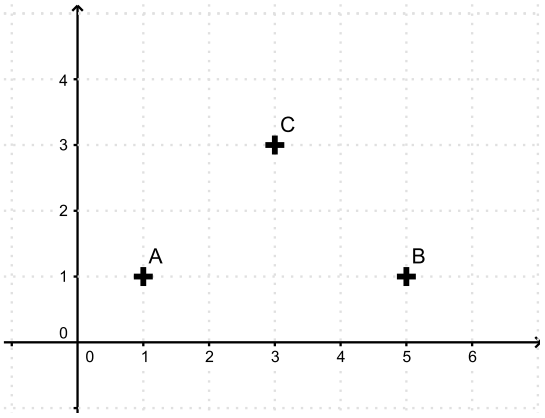
Example

standard input	standard output
2 12 4 1 5 4 1 9 7 2 1 1 1 5 1 3 3	YES NO

Note



The first testcase



The second testcase

Problem G. Youngling Tournament

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Yoda, the Grand Master of the Jedi Order, hit on the idea to hold a tournament among younglings. He has not chosen the date yet, but he has already decided on the format of the tournament.

There are N younglings studying in the Jedi Temple. Yoda can feel the Force inside younglings, moreover, he can represent the amount of the Force inside i-th youngling as the number f_i.

Therefore, the format of the tournament is the following. At first, Yoda makes the children stand in a row in order of non-increasing f_i. Then, the first youngling in the row competes against all the others united together. The combat is based on lightsaber battle and is very spectacular. However, Yoda knows that the result doesn't depend on anything but the Force inside competitors. The youngling wins if his amount of Force is not less than the total amount of the Force inside all his opponents. In that case, he is considered one of the winners. Otherwise, he loses. Anyway, after that he is removed from the row, and the tournament continues. Again, the strongest (first in the row) youngling competes against all the others standing in the row in the same format, if he wins, he is also considered one of the winners. After that he is removed and the tournament continues in the same format until there is only one child in the row. He becomes one of the winners automatically and the tournament finishes.

Yoda wants to know the total number of winners. However, as the tournament is postponed again and again, the amount of the Force inside the younglings changes from time to time. Help Yoda to compute the total number of winners after each change.

Input

The first line of input contains a single integer N, the number of younglings in the Jedi Temple ($1 \le N \le 100\,000$).

The second line contains N integers $f_1, f_2, \ldots, f_N$ the amount of the Force inside the younglings ($1 \le f_i \le 10^{12}$).

The third line of the input contains a single integer M, the number of changes in the Force amounts of students ($0 \le M \le 50\,000$).

The next M lines contain information about the changes. The i-th of these lines describes the i-th change and contains two integers k and f_k^*, which mean that the amount of the Force inside the k-th youngling becomes equal to f_k^* ($1 \le k \le N$, $1 \le f_k^* \le 10^{12}$).

Output

Print $M + 1$ lines, each of them containing a single integer.

On the first line, print the number of winners if the tournament was held before all the changes. On line $(i + 1)$ for all $i > 0$, print the number of winners if the tournament was held after the first i changes.

Examples

standard input	standard output
3 2 1 3 3 1 3 2 7 3 5	3 2 3 2
7 2 14 14 15 5 2 5 5 5 2 4 12 5 4 3 10 7 9	4 3 3 3 3 4

Problem H. Garland Checking

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

This is an interactive problem.

When preparing to New Year Eve, Queen Amidala always checks her electric garland to ensure it works. The garland consist of n lamps which are numbered from 1 to n; there are $(n-1)$ pairs of them which are allowed to be connected with wires to form a special structure. Each wire in the structure connects exactly two different lamps, and all the lamps are connected together with the wires.

However, the garland is so huge that this year, the Queen decided not to build the whole structure at once. However, she still needs to check each wire. So, she will connect some of the wires, then check if electric current can pass from some lamps to some others, then disconnect some of the connected wires, connect other ones and so on. More formally, the Queen will perform a sequence of operations. Each operation will be one of the following:

- Connect two different lamps with a wire;
- Remove a wire that connects two lamps;
- Test if electric current can pass between two lamps by the wires that are currently connected.

In order to properly test the garland, she will always follow the garland structure. This means that she will only connect pairs of lamps which are allowed to be connected. Moreover, Amidala will connect and disconnect any of the $(n-1)$ allowed pairs exactly once.

The Queen was almost starting to check the garland, but she realized that not all her tests will be meaningful because some pairs of lamps will be disconnected at the time she checks them, and the current will not pass even if the garland is in fact working. She asked Anakin Skywalker to help in checking her plan of checking the garland. Anakin is sure that it's boring to proceed all the operations manually, so he asked you to write a program which will process all three types of operations described above; for each test operation, the program must answer whether the two lamps are connected by wires of not. Anakin shouldn't disgrace himself in Amidala's eyes, so don't let him down!

Interaction Protocol

At start, your program will be given an integer n ($1 \le n \le 100\,000$). After that, you have to process queries one by one. Each query will be entered on a separate line in one of the following forms:

- "`C` a b", where a and b are integers ($1 \le a, b \le n$, $a \ne b$). This query means that the Queen is connecting lamps a and b with a wire. There will be exactly $(n-1)$ such queries.
- "`D` a b", where a and b are integers ($1 \le a, b \le n$, $a \ne b$). This query means that the Queen is disconnecting lamps a and b. It's guaranteed that at the moment this query is entered, lamps a and b are connected by a wire; note that pairs (a, b) and (b, a) correspond to the same wire. There will be exactly $(n-1)$ such queries.
- "`T` a b", where a and b are integers ($1 \le a, b \le n$). This query means that the Queen is testing if electric current can pass between lamps a and b. You must print a line containing "`YES`" (without quotes) if lamps a and b are reachable from each other by wires, and "`NO`" otherwise. Do not forget to flush your output.
- "`E`", this means the end of checking, your program must immediately terminate after receiving this query.

There will be no more than 300 000 queries.

It is guaranteed that the queries will always follow the garland's structure, and each possible wire will be connected and disconnected exactly once.

Initially, all possible wires are disconnected.

Example

standard input	standard output
3 C 1 2 C 2 3 T 1 2 D 2 3 D 1 2 T 1 2 E	 YES NO

Problem I. Equipment Assembling

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Thousands of clones work at the biggest imperial factory which produces parts of stormtrooper equipment: blasters, body armors, helmets and so on. In the final step the equipment should be assembled using N pieces. Clones know M different methods of connecting one piece with another: the i-th method connects pieces a_i and b_i, and it takes t_i seconds.

When several pieces are connected together, they are considered a single unit, and it's no longer possible to connect any pair of them. Of course, it's not possible to connect pieces which belong to different parts of equipment, and it's always possible to assemble any part of equipment using these methods. A part is considered assembled if all the pieces of this part form a single unit, no matter how they are connected inside.

To increase productivity, it was decided that all parts have to be assembled in the fastest possible way. However, if the fastest way is not unique, then the resulting equipment can look differently. Two ways of assembling differ if the sets of applied methods of connecting differ. It's known that the army of clones looks awesome only if all the clones are identical. Because of that, the headmaster of the factory decided to retrain clones to apply some methods in different time so that the fastest way of assembling will be unique. Let's denote the new number of seconds needed to apply i-th method of connecting by t_i^* (note that for some i, the values of t_i and t_i^* may be equal). The time spent on method's application has to be an integer number of seconds and could not be negative.

However, retraining is a complex process, and it will take one day to change the time one method takes by one second. The headmaster asked you to compute the minimal possible number of days he needs to spend on retraining clones in order to make the fastest way of assembling unique.

Input

The first line of input contains two integers N and M: the initial number of pieces and the number of connecting methods ($1 \le N \le 20$, $0 \le M \le 1000$).

The next M lines contain information about methods of connecting two pieces. The i-th of them contains three integers a_i, b_i and t_i: the numbers of pieces that can be connected and the time this method takes ($1 \le a_i, b_i \le N$, $a_i \ne b_i$, $1 \le t_i \le 10^6$).

Output

On the first line, print a single integer: the minimal possible number of days the headmaster needs to retrain clones.

On the next M lines, print the descriptions of the new connecting methods: on the i-th of these lines, print three integers a_i, b_i and t_i^*. The new connecting time t_i^* must be a non-negative integer not greater than 10^9. The methods must be printed in the same order as in the input. It's guaranteed that there exists a solution that satisfies all the above constraints.

Examples

standard input	standard output
3 3 1 2 2 1 3 1 2 3 2	1 1 2 2 1 3 1 2 3 3
8 10 1 2 3 1 4 3 2 4 3 2 3 4 4 3 5 5 8 1 7 8 1 5 6 2 7 6 2 8 6 3	2 1 2 3 1 4 3 2 4 4 2 3 4 4 3 5 5 8 1 7 8 1 5 6 2 7 6 3 8 6 3