Problem 1. Painting roofs

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

The King of Berland loves order in everything. For instance, the capital of Berland as seen on the map is a rectangular field of cells, that correspond to city quarters.

He recently has issued a decree stating that all roofs in a quarter must be painted the same color — either red or blue. The capital of Berland is expecting a commission which will check the whole city for compliance, moving from one quarter to an adjacent by a side quarter. For a successful check, the commission must be able to reach any quarter from any other.

The Minister of Finance found out that the commission has a curious demand. They cannot move to an adjacent quarter if its roofs have the same color as roofs of the quarter they are currently in. Consequently, some quarters will have to repaint all their roofs to pass the check. The Minister of Finance wants to save money. He's asking you to find the minimal number of quarters where all roofs must be repainted.

Input

The first line of the input file contains two integers m and n — the number of lines and columns in the city map ($1 \leq m, n \leq 1000$). Next come m lines consisting of n symbols, with each symbol being either '1', or '2'. The symbol '1' means that all roofs in the corresponding quarters are blue, and the symbol '2' means that the roofs are red.

Output

In the output file, print a single integer — the minimal number of quarters where roofs must be repainted.

Example

`input.txt`	`output.txt`

`1 4`
`2211`
	`2`

Problem 2. Symmetric matrix

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

In a square matrix with n rows, all elements are integers. You can swap two elements of the matrix at one step. Find out the minimal number of steps necessary to obtain a symmetric matrix from the initial one. A symmetric matrix is a matrix with the same element at the intersection of the ith row and jth column as that at the intersection of the jth row and the ith column for any i, j.

It is guaranteed that in this matrix, n elements occur only once, and each of the rest occurs twice.

Input

The first line of the input file contains an integer n — the number of rows in the matrix ($1 \leq n \leq 500$). The following n lines describe the rows of the matrix. Each of them contains n space-separated integers, which are not greater than 10^9 in absolute value.

Output

In the first line of the output file, print a single integer m — the minimal number of steps to obtain a symmetric matrix. Next, print an example of m such steps. For the ith step, print four integers a_i, b_i, c_i, d_i into a separate line, which mean that the element located in the a_ith row and the b_ith column must be swapped with the element in the c_ith row and the d_ith column.

Examples

input.txt	output.txt

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

3
1 4 -3
4 2 5
6 6 5
	2
	3 3 3 2
	3 3 1 3

Problem 3. Game map

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	3 second
Memory limit:	256 megabytes

Vasya still likes to play computer games, and he's still employed as a game tester. He's lost all hope of doing anything remotely interesting at work. To realize his creative potential, he is cooperating with his colleagues Petya and Kolya to work on a hobby project. Petya has told Vasya that there are a number of great indie games out there, and even a small team can do well in that business. He's not being entirely honest here — he hasn't told the guys about what percentage of those indie games gains any popularity or how many of those projects are completed at all.

The new game will be in the popular roguelike genre. The bearded Kolya keeps correcting Vasya: it's roguelite or roguelikelike. One of the most important components of a roguelike game is the randomly generated gameworld. The new adventure at each launch is the key to making "Losing is fun" motto work. The friends are working now on world map generation for their game.

The game takes place in caves. The world map is a rectangular field of cells. Each cell is either traversable, containing a level that must be passed by the player, or non-traversable and does not contain anything. The player begins the game at the top left corner cell, which must be traversable. Having finished a level, the player can move down or right to an adjacent cell, if that cell is traversable. While in the last row or column of the field, the player can leave the field. This ends the game, and the player wins.

For the purpose of balance, Petya has placed additional limitations on the map:

1. The player can reach each traversable cell from the starting cell though only one path.
2. The player can finish the game starting from any traversable cell, however, there can be several ways to do that.

Kolya is the chief programmer of the game and he will write the generation code. He has noticed that if the field is generated randomly, it almost never satisfies the criteria. For this reason, he's randomly generating only some of the field cells, leaving the rest in an «indefinite» state. Next he must run an algorithm that defines the contents of the remaining cells in such a manner that the resulting map satisfies all of the conditions. To increase the amount of content in the game, the algorithm must maximize the number of traversable cells.

The trouble is, Kolya is not a big expert on algorithmic programming and cannot implement the last step on his own. Help the friends make their dream come true!

Input

The first line of the input file contains two integers defining the size of the game field: H — the number of rows and W — the number of columns ($1 \le H, W \le 18$).

It is followed by the field contents as H lines with W symbols in each. Traversable cells are defined by the period symbol `'.'` (ASCII 46), non-traversable — by the letter `'X'` (ASCII 88), and indefinite — by the question mark `'?'` (ASCII 63).

Output

If it is impossible to define the game field to a correct state, print the number -1 in the output file.

Otherwise, in the first line of the output file print an integer K — the number of traversable cells in the define game field. This number must be as large as possible. Next, print the defined game field in the same format as that used in the input data — naturally, without the question marks.

If there are several optimal solutions, print any of them.

Examples

input.txt	output.txt
4 6 ??X??? .????? ????.. ?X?X?.	13 .XXXXX .....X .X.X.. .X.XX.
3 3 ??X ?X? X??	-1

Problem 4. Cloud computing

Input file:	`standard input stream`
Output file:	`standard output stream`
Time limit:	2 seconds
	5 seconds (for Java)
Memory limit:	256 megabytes

Cloud computations are gaining popularity as a powerful and versatile tool. However, they are seriously flawed: processing your data on a remote computer puts your information safety at risk.

Vanya works in an Organization which implemented cloud computing for calculating order statistics of arrays. An order statistic of an array for a specific k is the value of the element, which is kth in the array, if the array is sorted.

However, the array which requires order statistics is extremely classified. The only thing known about it is that all its elements are different. With this in mind, Vanya came up with the following scheme: the array is stored on the Organization's server, and the cloud server performing the order statistics calculations can access the Organization's server to get the results of the comparison of two elements of the array. In this manner, the cloud server can define the position of the kth order statistic, and the classified array is never revealed to the cloud server. This produces another problem: the number of requests from the cloud server to the Organization's server should not be exceedingly large.

In particular, Vanya decided to limit the calculations of the second order statistics to no more than $N + 20$ requests, where N is the size of the array. Help Vanya implement an algorithm of finding the second largest element of the classified array, such that it complies to this limitation.

Interaction Protocol

This is an interactive problem. Instead of file input-output, you will be working with a special program — the interactor, using the standard input-output streams. When your program starts, it feeds the integer N into the standart input stream — the size of the array($2 \leq N \leq 10^5$).

Next, your program must send requests to the standard output stream.

Each request must consist of a single line with a question mark («?») followed by two different, space-separated numbers I and J — the indices of those elements of the array that must be compared ($0 \leq I,\ J < N$). In return, you get a string containing the «less» symbol («<»), if the Ith element of the secret array is smaller than the Jth one, or the «more» symbol («>»), if the Jth element of the secret array is smaller than the Ith one.

When your program defines the position of its second order statistic in the classified array, it must print a line with that position following an exclamation mark («!») and separated by a space symbol.

Make sure that you arer printing the line break symbol and clearing the output stream buffer (the `flush` command of the language) after every printed line. Otherwise the solution can receive the Timeout verdict.

Pay attention: interactor could alter the contents of the classified array in runtime if all the previously given answers stay true.

Example

For ease of reading, commands in the example are separated by empty lines.

standard input stream	standard output stream
4	
	? 0 1
<	
	? 0 2
<	
	? 0 3
<	
	? 3 2
>	
	? 2 1
>	
	! 1

Problem 5. Cone lights

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

During his failed attempt at crossing the border, Ostap was robbed down to his flesh. But the damn criminals didn't get all of the bling — the commandor trampled the bishop's cross and cigar cases in a snowbank, hiding them securely. Ostap wants to bribe the border officials with the Order of the Golden Fleece, which he has hidden from the burglars; that will lead him to the treasures. He has planned a search route through the snow — a polyline on the plane XY. The treasures are hidden in one of the points of the polyline. The problem is, the search will be held in complete darkness, and treasures aren't all that easy to find.

He has made a deal with the chief of border patrol — he will lit the search route with projector lights. Each point of the polyline must be lit with no less than K lights. All lights are turned on at once and have the same coverage angle, which we can change.

For each light, we know its coordinates and the vector along which the light is pointed. Our task is to find the minimal possible angle of lighting (the narrower the angle, the smaller the fee) sufficient to find the treasures. The angle must be identical for all lights.

More formally, if the light is in the point P with the coordinates P_x, P_y, P_z, pointed in the direction L with the coordinates L_x, L_y, L_z and has a lighting angle of φ, then the point Q will be lit when and only when the angle between the vectors $\overrightarrow{PQ}$ and $\overrightarrow{L}$ is no greater than φ.

Input

The first line of the input file contains thee space-separated integers N, M and K — the number of segments in the polyline, the number of available lights and the requred number of lights, respectively ($1 \le N \le 100, 1 \le M \le 100, 1 \le K \le M$).

The following $N + 1$ lines describe the coordinates of each of the vertices of the polyline. Each i-th line contains two integers X_i and Y_i — the coordinates of the i-th vertex of the polyline. For all vertices, the z-coordinate equals 0. The polyline can intersect and osculate itself.

Next come M lines that describe the lights, one light per line. A description of a light consists of six integers P_x, P_y, P_z, L_x, L_y, L_z — the coordinates of the light and the coordinates of its vector, respectively ($P_z > 0$). It is guaranteed that at least one of the coordinates L_x, L_y, L_z is not zero.

All vertices of the polyline are in different points. Projector positions could match. All coordinates are no greater than 10^3 in absolute value.

Output

In the output file, print a single real number — the minimum possible lighting angle in degrees, such that all points of the route are lit with no less than K lights.

The relative or absolute error of the answer should not be greater than 10^{-6}.

Example

input.txt	output.txt

2 5 2
2 3
8 3
6 6
2 3 3 0 0 -1
5 3 3 0 0 -1
8 3 3 0 0 -1
4 7 1 1 0 -1
6 7 2 0 0 -1
	45.00000000

1 1 1
2 0
2 2
0 0 1 0 0 1
	116.565051177077989

Example explanation

In the first sample almost all of the projectors are pointed in the down direction, and if the angle is 45 degrees the circle lit with a projector has radius equal to the height the projector is located at. For the first segment to be lit twice the first three projectors are enough. The second segment is lit with the last two projectors.

In the second sample the projector is pointed in the up direction, so the angle more than 90 degrees is needed. The main aim is to lit the point (2, 0). An angle of $90^\circ + \arctan \frac{1}{2}$ is needed for that.

Problem 6. Tote

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	2 seconds
Memory limit:	256 megabytes

The first thing Ostap did upon his arrival to Rio was buying white slacks. The second thing was wasting all the money he had smuggled in the local football tote.

There are n matches played in the state championship, each can have one of the three possible outcomes: win, tie or loss.

The list of all matches is given in the tote ticket. Immediately after buying a ticket, the buyer must mark for every match which outcomes he bets on. For each match, he may either mark one outcome (so-called "single"), or mark two outcomes ("double"), or even mark all three outcomes ("triple").

The tickets of various types are sold. The type of the ticket defines its cost, and two numbers i and j. The buyer must put exactly i doubles, exactly j triples, and exactly $n-i-j$ singles in the ticket. How to distribute singles, doubles, and triples across matches, and which outcomes to bet on — that's what the buyer can choose.

If the buyer guessed the results of all n matches correctly, then he gets P coins for the winning ticket. If he was not lucky enough, then he gets nothing for it. The buyer guesses the result of the match if and only if his ticket has a mark on the actually happened outcome for the match.

Thanks to his acquaintance with Velial, the demon of gambling, Ostap knows that the outcome of every match is determined randomly. Ostap even managed to learn the probabilities of all outcomes! Moreover, he knows that the outcomes of the n matches are determined independently of each other.

Ostap has only S coins left, which he can spend on tickets. Being a regular customer, Ostap can buy as many tickets of each available type as he likes, as long as he has enough coins to cover the cost. All that is left is to distribute the money wisely in such a manner that the expectation of prize money is maximum possible. Note that Ostap maximizes prize money he will get back, and it does not matter how much the tickets will cost him.

Input

The first line of the input file contains four integers: n — the number of matches, k — the number of ticket types, S — how much coins Ostap has, and P — how nuch coins are given for a winning ticket ($1 \le n, k \le 100$, $1 \le S \le 10^6$, $1 \le P \le 10^{18}$).

It is followed by n lines with probabilities of match outcomes. ith of these lines contains three real numbers x_i, y_i and z_i with at most 8 digits after decimal dot — the probability of win, tie, and loss respectively in the ith match ($0 \le x_i, y_i, z_i \le 1$, $x_i + y_i + z_i = 1$).

Next come k lines, describing which types of tickets are available for sell. Each line has three integers: i and j — how much doubles and triples respectively must be marked in a ticket of this type, and c_{ij} — the cost of one ticket of this type ($0 \le i, j \le n$, $i + j \le n$, $1 \le c_{ij} \le 10^6$).

Output

Print a single real number, being the maximum expected value of the prize money.

Absolute or relative error of your answer must not exceed 10^{-9}.

Examples

input.txt	output.txt

2 2 10 10
0.5 0.3 0.2
0.8 0.2 0
0 0 4
1 0 6
	10.4

3 4 1000 10000
1.0 0 0
0.5 0.5 0
0.3 0.4 0.3
0 0 100
1 0 200
1 1 300
1 2 400
	32000

Example explanation

In the first sample, tickets of the first type allow only singles. If we mark win in both matches, then the ticket wins with 40% probability. Hence, the expectation of prize money won for every ticket of this type is 4.

In the tickets of second type, there must be one double. The optimal usage of such ticket is to mark double in the first match, betting on win and tie, and mark single in the second match, betting on win only. Then the ticket wins with 64% probability, which gives 6.4 coins of prize money on average.

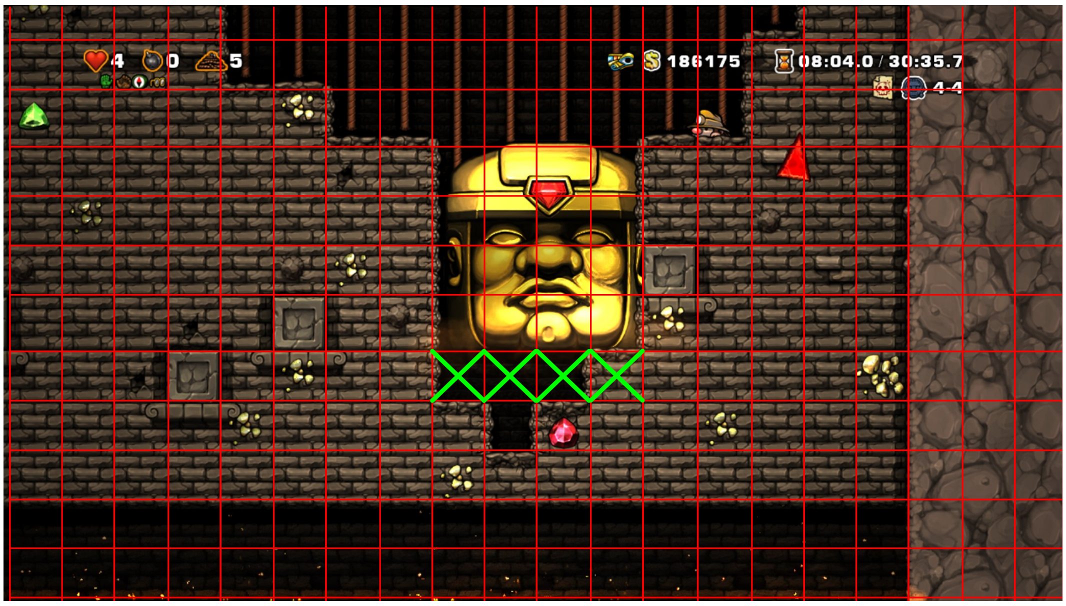
Ostap has only 10 coins left. Winning one ticket of each type is the best he can do.

Problem 7. Olmec

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	3 seconds
Memory limit:	256 megabytes

Olmec is the boss in Spelunky, a computer game. You must beat Olmec in the last level of the game. Olmec moves in the game field, following the player. Every time he moves, he jumps high above and then falls down vertically. When the player is underneath the soaring Olmec, he drops down especially hard, intending to crush the player. The impact destroys the ground where he lands. The only way to beat Olmec is to make him drill a hole down to the very bottom of the level, which is filled with hot lava, and make him fall there.

The Spelunky game happens on a rectangular grid, which is a side-view of the action. Each cell of the grid is either empty or filled with ground. Olmec is a $K \times K$-sized square, although his height is irrelevant to us.



Assume that Olmec's strike works as follows. Soaring high above the ground, where all cells are empty, Olmec chooses a vertical to fall along, thus defining his horizontal position. Next, he falls vertically until he collides with a ground cell. At the impact, K adjacent cells underneath Olmec are affected and become empty. After that, Olmec rises back into the air and prepares for the next strike.

The picture shows the moment of collision in the Olmec's strike with $K = 4$. The affected cells are crossed out. All these cells will become empty as the result of the impact, even though only one of these cells had ground before strike.

You are given a state of the game field and a set of rectangular queries. Assume that we can completely control Olmec's actions and can make him perform any predefined series of strikes. For each query, define the minimal number of strikes necessary to empty all cells of the given rectangle.

Input

The first line defines four integers: H — height of game field, W — width of game field, K — size of Olmec, and Q — number of queries ($1 \le H \le 12$, $1 \le K \le W \le 10^5$, $1 \le Q \le 10^5$).

The following H lines describe the game field, with W symbols per line. The letter `'X'` (ASCII 88) defines a cell with ground, and the period symbol `'.'` (ASCII 46) defines an empty cell. Rows of the game field are provided from top to bottom.

The remaining Q lines list the queries, one per line. Each query contains three integers: D — rectangle depth, L — number of first column in the rectangle and R — number of last column in the rectangle ($1 \le D \le H$, $1 \le L \le R \le W$). A cell located in the row r and column c belongs to the defined rectangle if $r \le D$ and $L \le c \le R$. It is guaranteed that the rectangle width is not smaller than the width of Olmec (i.e. $R - L + 1 \ge K$).

Consider the queries to be theoretical: no real actions are performed in the field, and each query is analyzed independently from the others.

Output

In the output file, print Q integers, one per line. Each number is the minimal number of strikes necessary to completely empty the corresponding rectangle. Answers to queries must be printed in the same order in which they are listed in the input file.

Example

input.txt	output.txt
`12 20 4 5`	`3`
`XXXX...........XXXXX`	`1`
`XXXX..........XXXXXX`	`0`
`XXXXXX........XXXXXX`	`7`
`XXXXXXXX....XXXXXXXX`	`41`
`XXXXXXXX....XXXXXXXX`	
`XXXXXXXX....XXXXXXXX`	
`XXXXXXXX....XXXXXXXX`	
`XXXXXXXX...XXXXXXXXX`	
`XXXXXXXXX.XXXXXXXXXX`	
`XXXXXXXXXXXXXXXXXXXX`	
`.................XXX`	
`.................XXX`	
`12 9 12`	
`3 6 11`	
`1 9 13`	
`5 4 10`	
`10 1 20`	

Example explanation

The example is the same game field as the one shown in the picture in the problem description.

Problem 8. Game of strings

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	3 seconds
Memory limit:	256 megabytes

Alisa, Boris and Konstantin are playing a game of strings. The rules are the following:

1. Alisa chooses a string A.
2. Boris chooses a string B.
3. Konstantin, who is the game moderator, chooses a positive integer k.
4. In the string A, a substring X of the length k is selected randomly. The starting position of the substring is selected equiprobably among all possible cases.
5. In the same manner, a random substring Y of the length k is selected in the string B.
6. The game outcome depends on which of the strings is lexicographically smaller. If the substring X is lexicographically smaller than the substring Y, Alisa wins. If the substring Y is lexicographically smaller than the substring X, Boris wins. If the substrings are equal, friendship wins.

Alisa and Boris have already come up with their strings A and B. Konstantin is curious: what is the probability of each of the outcomes? Calculate these probabilities for all reasonable values of the number k.

Input

The first line of the input file contains a string A, and the second line contains a string B. The string A consists of n symbols, and the string B consists of m symbols ($1 \le n, m \le 2 \cdot 10^5$). Strings contain only lower case Latin letters.

Output

In the output file, print $\min(n, m)$ lines, with three real numbers in each. The results for the case when Konstantin chooses a number k, must be placed in the kth line. The first number in the line defines the probability of Alisa winning, the second is the probability of friendship winning, and the third is about Boris winning.

The deviation of each printed number from the correct value should not be greater than 10^{-12}.

Example

input.txt
abac ababa

output.txt

0.2 0.4 0.4
0.33333333333333 0.333333333334 0.333333333333333
0.1666666666666 0.3333333333333 0.500000
0.5 0 0.5

Problem 9. Steve's perfectionism

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Steve is a hopeless perfectionist. He's been stumbling upon arrays of positive integers, but he doesn't always enjoy what he's seeing. Steve likes an array if and only if it is strictly ascending. Arrays that Steve doesn't like must be changed asap, otherwise Steve will be disappointed. The new array will satisfy Steve only if its initial length is retained, and its elements are divisible by the corresponding elements of the old array. Note that an array can be satisfactory from the start. In this case, it can be left alone (or not, if you want change).

Make Steve happy! Write a program that can instantly produce new, likeable arrays.

Input

The first line of the input file contains a single integer n — the number of elements in the array that Steve has laid his eyes on ($2 \leq n \leq 1000$). The second line of the input file contains n space-separated positive integers a_i ($1 \leq a_i \leq 10^6$).

Output

In the output file, print n positive integers a_i ($1 \leq a_i \leq 10^9$) — the new array, which Steve will like.

There can be several solutions: print any of them.

Example

input.txt	output.txt
7 1 10 5 4 3 2 7	1 10 15 20 30 32 35

Problem 10. Wooden pipeline

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	3 seconds
Memory limit:	256 megabytes

In a land far, far away there lived a wise king who ruled N cities. The king decided to improve life in his kingdom and made the gentry get busy with innovations, modernization, and nanotechnologies. The gentry came up with an innovative pipeline network. Since they haven't really mastered the nanotubes yet, the pipelines are made of wood.

The wooden pipelines connect all cities into a single network, where any city can be reached from any other city. The network consists of $N - 1$ pipelines. Each pipeline leads directly from one city to another without any forkings. The additional pipelines were planned according to the original costs estimate, but in the end they ran out of wood.

We know the capacity of each pipeline in each direction, i.e. the amount of fluid which can pass through it in a unit of time. The capacity of a pipeline in the opposite directions can be different, due to the (in)famous craftsmanship of the kingdom.

The king is deeply saddened, contemplating the fruit of his denizens' labor. He cannot think of a liquid to pump through the pipelines. Milk would go sour, and the kingdom is not that rich in mead. On the other hand, they could pump water, in case there is drought somewhere. The king wants to know the efficiency of the pipeline system in case of a drought.

Assume there is a drought in the city u. This splits all other cities into two types: terminal and transitional cities. A city is transitional if it has a pipeline leading to a city which is more distant from the city u, with the distance calculated along the pipelines. All other cities are terminal. You can take water from all terminal cities in any volume and pump it along the pipelines to the city u. You cannot take water from transitional cities.

Define the maximum volume of water per unit of time that can be pumped through the pipelines from the terminal cities to the city u. Drought can affect any of the cities, so calculate the answer for each case of u.

Input

The first line of the input file contains a single integer N — the number of cities ($1 \le N \le 3 \cdot 10^5$). The remaining $N - 1$ lines describe the pipelines, one per line. Each pipeline is described by four numbers: a — the number of the city where the pipeline starts, b — the number of the city where the pipeline ends, C_{ab} — pipeline capacity in the direction from a to b, C_{ba} — pipeline capacity in the direction from b to a ($1 \le a \ne b \le N$, $1 \le C_{ab}, C_{ba} \le 10^5$).

It is guaranteed that you can reach any town from any other town along the pipeline system.

Output

In the input file, print N integers, one per line. Each kth number defines the maximal volume of water per unit of time that can be pumped into the city with the number k in case it is stricken by drought.

Example

input.txt	output.txt

5
1 2 2 4
5 2 2 6
2 3 2 3
3 4 5 5
	4
	7
	7
	2
	5

Problem 11. Cubic polynomials

Input file:	`input.txt`
Output file:	`output.txt`
Time limit:	1 second
Memory limit:	256 megabytes

Two old pals, Cardanieux and Ferrarineux, love mathematical competitions: one of them emails the other a mathematical problem, with week to solve it. If the friend fails to cope with the problem in a week, he loses.

Ferrarineux came up with yet another problem and has already sent it to his friend. Cardanieux has just read the new challenge — he's got a big pile of cubic equations of the shape $ax^3 + bx^2 + cx + d = 0$, and the challenge is to multiply all this polynomials and find an integer, such that it is root of the largest multiplicity of the product. Cardanieux is very worried: he has lost three times in a row. This time, he just cannot afford to lose. He is asking you to write a program which will find a solution to the challenge, so that he can disappoint his friend with a lighting-fast correct answer.

Input

The first line of the input file contains a single integer n — the number of the equations ($1 \le n \le 10^5$). It is followed by n lines, with each ith line containing four integers a_i, b_i, c_i and d_i, which are all non-zero — the coefficients of the ith equation $a_ix^3 + b_ix^2 + c_ix + d_i = 0$ ($0 \le |a_i|, |b_i|, |c_i|, |d_i| \le 10^9$).

It is guaranteed that all numbers a_i are non-zero.

Output

If none of the equations has an integer root, print a single line `NO`. Otherwise, in the first line of the input file, print the `YES`. In the second line, print two space-separated integers — the first one is the root of the largest multiplicity of the product of these equations, and the second one is the multiplicity.

If there are several solutions, print any of them.

Examples

input.txt	output.txt

2
1 3 3 1
2 2 0 0
	YES
	-1 4

1 1 1 1 2	NO

Example explanation

In the first example the product of the polynomials equals

$$(x^3 + 3x^2 + 3x + 1) \cdot (2x^3 + 2x^2) = 2x^2(x + 1)^4$$

The number 0 is a root of multiplicity 2 and the number -1 is a root of multiplicity 4.

In the second example there are no integer roots.