Problem A. Anthem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Dwarf the Piper has been playing the Dwarf National Anthem (DNA) on his giant pipe organs every morning for centuries. But now, he has to move to a new cave with limited space, where his original pipe organs no longer fit.

The DNA is a sequence of notes (letters from `a` to `z`), and each pipe of the organ can play a specific note. Dwarf the Piper must build a compact organ with the smallest number of pipes that can play the entire DNA.

Organ pipes are stored in a row. To play the DNA, Dwarf the Piper can start at any pipe. Once he has blown the pipe making the appropriate note, he can stay in the same place, move to the next pipe to the left, or move to the next pipe to the right. He can stop playing at any place. Help the Piper to build the shortest pipe-organs that can play the DNA.

Input

The first line of input contains an integer number N representing the number of notes in the DNA. The second input line contains the DNA as a string of N lowercase letters.

Limits $1 \leq N \leq 500\,000$.

Output

The first line of the output should contain an integer K which is the smallest number of pipes in an organ that can be used to play the DNA. The second line of the output should contain the description of such organs: a string of K lowercase letters which are the notes assigned to the pipes from left to right. If there are many possible shortest organs, your program can print any of them.

Example

standard input	standard output
14 ecerrcwrwcwror	7 cercwro

Problem B. Bad Digits

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

As the end of the year approaches, the dwarf magistrate decides that their village should switch to a positional number system with a new base of N. For bases greater than 10, the dwarfs use digits `0` through `9`, followed by the uppercase English letters from `A` through `Z`. But the cause of all the problems is not the magistrate, but the dwarf elders, who, after consulting ancient tomes, decided that some of these digits are forbidden. Notably, the digit `0` is not on the list – after all, the dwarfs consider it a sacred symbol of balance.

The numbering system is a cornerstone of dwarf orderly society: the houses in each village are numbered with smallest distinct positive numbers (using the correct base N and not using forbidden digits). The magistrate wants to know what is the largest house number that will be used in the village.

Input

The first line of the input contains an integer T, the number of test cases (the number of villages). The following lines contains the description of the test cases.

The first line of test case contains three integers N, K, and M separated by single spaces, where N is the base for positional notation, K is the number of houses in a village and M is the number of forbidden digits. The second line contains the list of M distinct forbidden digits (in base N), separated by single spaces. If $M = 0$, the second line is empty.

Limits $1 \le T \le 10\,000$, $2 \le N \le 36$, $0 \le M \le N - 2$, $1 \le K \le 10^{18}$.

Output

For each test case, print one line containing the largest house number in the village (in the positional system with base N).

Example

standard input	standard output
6 3 5 1 1 9 7 1 7 2 100 0 7 123456789 3 3 6 1 16 123456789 3 1 A F 36 123456789 5 A X Z 2 4	202 8 1100100 25224550520222 2E878582 6C0503

Problem C. CERC Plaques

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

CERC 2024 in Wroclaw is approaching, and all hands are on deck to help with organization. Even Wroclaw's dwarfs are helping! As a member of the organizing committee, each dwarf was given a plaque, so everyone can identify him. Dwarfs are very creative creatures, so each of them came up with an unique *nickname* that they want to be displayed on the plaque. A nickname is correct if its first four letters match with first four letters of the dwarf's name (we treat lowercase and uppercase letters as different). For example, dwarf Mathew can have a plaque that reads `Mathy`, but he can not have a plaque that reads `Matty` or `MATHY`.

Dwarf the Sloppy printed the plaques. Since he is, well, sloppy, he made no notes which plaque is whose. To make things worse, some of the plaques may contain errors. Help Sloppy to figure out which plaque belongs to whom: You will be given the list of all the dwarfs' names and the list of all nicknames on the plaques. Write a program which decides whether there exists an assignment of the plaques to the names such that the plaque contains a proper nickname of the assigned name. If such an assignment exists, your program should also print it.

Input

The first line of the input contains a single integer N, the number of dwarfs. Each of the following N lines contains a single dwarf's name, which is a string of lowercase and uppercase English letters.

Each of the following N lines contains a single nickname written on a plaque, which is a string of lowercase and uppercase English letters.

Limits $1 \leq N \leq 100\,000$, each name and nickname contains at least 4 and at most 400 000 letters, the sum of all lengths of the names and the sum of all lengths of nicknames does not exceed 400 000, it is guaranteed that no name and no nickname appears twice.

Output

The first line of output should contain a single word – `YES` if the assignment described above is possible and `NO` if there is no such an assignment.

If the answer is `YES`, then the following N lines should contain the correct assignment: Each of the lines should contain the name of the dwarf and the nickname assigned to him, separated by a single space. If there are many possible assignments, print any of them.

Examples

standard input	standard output
4 Slopy Mathy Thinky Cody Thinky Math Slopppy Codythesecond	YES Cody Codythesecond Mathy Math Slopy Slopppy Thinky Thinky
3 Writy Buggy Solvy Bogg Write Solvy	NO

Problem D. Divisors

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Ori Badpun, a know-it-all-dwarf, taunts you with a puzzle: Algorist, eh? Knows all about *divide and conquer*, or *divide et impera*, paradigm, eh? Or perhaps you got it backwards?! How about textitimpera et divide or *conquer and divide*? Let a *permutational divisor* of a number N be defined as a proper divisor (i.e., less than N) whose digits are a permutation of the digits of N with leading zeros not allowed. Got it? I will test your skills in T trials. In each, you will get a positive integer N, and you are to tell me how many permutational divisors it has. So, you got it now?

Input

The first line of the input contains the number of test cases T.

Each of the following T lines contains one natural number N, for which the answer needs to be determined.

Limits $1 \leq T \leq 100\,000$, $1 \leq N \leq 10^{18}$.

Output

For each number from the input, print a single integer on a separate line, indicating how many permutational divisors the given number has.

Example

standard input	standard output
4 7 31 90 370521	0 0 0 1

Note

Number 370521 is divisible by 123507.

Problem E. Expression

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Dwarfs' legendary machinery construction skills allowed them to build mechanical computing machines predating Babbage's analytical engine. Recently though, a trend of minimalism has gained popularity, leading some to suggest that these elaborate machines might be swapped for simpler ones with reduced sets of instructions, which should be much smaller, cheaper, and perhaps also significantly faster in operation. So far, the dwarfs were unable to determine what is the minimal set of operations that suffices to perform all computations they deem important, and to resolve this issue, they have decided to perform numerous tests. You are to solve some of them, and your goal is to rewrite, if possible, given expressions into equivalent ones using only a specified subset of operators.

You are given an expression consisting of binary operators `min`, `max`, `<=`, `<` and variables (the first 10 letters of the English alphabet). A valuation of the variables from the set S is defined as assigning each element of S a value of 0 or 1. We say that two expressions are equivalent if both of the following conditions are satisfied:

- the sets of variables occurring in the expressions are equal – let us denote this set by S,
- for every valuation of S, the expressions with substituted values evaluate to the same value.

Your task is to create an expression consisting solely of the operators `min` and `<=` that is equivalent to the input expression, or to state that such an expression does not exist.

Input

The first and only line of standard input contains an expression. It is an `expr` in the following grammar.

- `var := a | b | ... | j`
- `op := expr <= expr | expr < expr | min expr expr | max expr expr`
- `expr := var | (op)`

Limits Given expression consists of at most 200 binary operations.

Output

If a solution exists, print `YES` on the first line of output, and on the second line any equivalent expression consisting of at most 40 000 binary operations. Otherwise, print a single line containing `NO`.

Note: It can be proven that if a solution of any size exists, there also exists one consisting of at most 40 000 operations.

Examples

standard input	standard output
`h`	`YES` `h`
`((max a a) <= b)`	`YES` `(a <= b)`
`((max a b) < (min a b))`	`NO`

Problem F. Flats

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

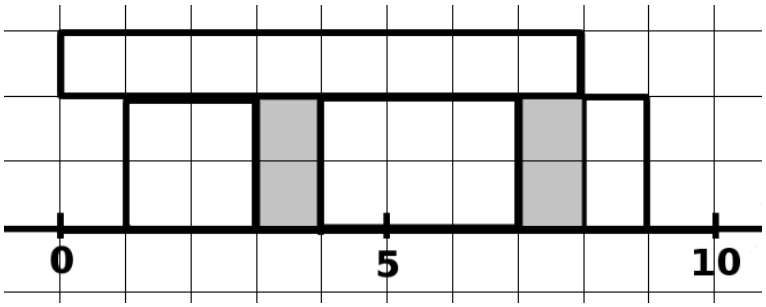
Dwarf underground city faces housing problems. Dwarf the Builder proposed a radical idea: they will use an ancient, seeming infinite and straight Great Divide ditch and throw the bricks into it. Then each enclosed space will be a new dwarf flat!

The Great Divide ditch has constant width but is very deep and seemingly infinite in both directions. To simplify the process, all used bricks will have the same width as the Great Divide (and varying length and height). When a brick is thrown to the ditch, its height is oriented vertically and width matches the ditch width. The brick will travel down until it reaches the ditch bottom (which is initially flat) or earlier-thrown brick. Edge-to-edge or side-to-side contact do not stop the bricks from falling, as the sides of the brick slide down while touching previous bricks. Also, the bricks do not change their orientation during the movement and after stopping.

We assume that the ditch is deep enough so that all bricks will fit into it. It is also long enough so that all bricks fit into it and no brick will ever reach the end of the ditch (in fact, no dwarf has ever reached it). The ditch is well-measured: it has a mark for each integer dwarfmeter from the origin.

A flat is made by a space (i.e. no brick) with non-zero volume that is fully enclosed by bricks (or ditch walls). Flats are considered separate, if a dwarf cannot move from one to another (and dwarfs have a volume). So flats with just "touching corners" are considered separate.

Help the Builder in promoting the idea and compute, how many flats he creates using his approach (and given bricks initial locations).



For simplicity we are ignoring the depth and plot the Great Divide ditch, bricks and flats in $2D$. Figure above presents the sample test. Two flats counted in the answer are marked grey.

Input

The first line of the input contains a single integer N, the number of bricks.

The next N lines describe bricks in the order of being thrown to the ditch. Each line contains three integers l, r, h, describing the coordinates of the left and right edges of the brick (so the brick has length $r - l$) and its height.

Limits $1 \le N \le 200\,000$, $0 \le l_i < r_i \le 200\,000$, $1 \le h_i \le 10^9$.

Output

Print a single non-negative integer – the number of flats created after all bricks have fallen.

Example

standard input	standard output
4 1 3 2 4 7 2 0 8 1 8 9 2	2

Problem G. Groceries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

In the dwarfs' underground city there are N houses, numbered from 1 to N clockwise. The houses are connected by one circular, bidirectional road. Distances between each two consecutive houses are known, each such distance is a positive integer.

The dwarfs love to visit their neighbours: a dwarf living in house i visits dwarfs living in houses $i+1$ (or 1, when $i = N$) and $i-1$ (or N, when $i = 1$). However, they do not like to come empty-handed, so after leaving their house and before arriving at their neighbour's house, they need to visit a grocery store to buy appropriate products. The visited grocery can be further away or in the other direction than the neighbour's house and the dwarfs can turn back at any spot on their way.

Dwarf the Planner wants to modernize the city by demolishing all old grocery stores and building K new ones, so that the maximal distance travelled by a dwarf when visiting a neighbour is minimal possible. New grocery stores can be built at arbitrary places on the circular road, not necessarily at the same locations as the houses and not necessarily at integer distance from an existing house. Help the Planner in this task and write an algorithm which computes, for the optimal location of K grocery stores, the maximum distance travelled by a dwarf who visits his neighbour and a grocery store on the way.

Input

The first line of input contains two integers N and K, representing the number of houses and the number of grocery stores to be built, respectively.

The second line of input contains N positive integers $d_1, \ldots, d_N$, representing the distances along the road in clockwise direction between houses i and $i+1$ for $1 \le i < N$ and between houses N and 1 for $i = N$.

Limits $2 \le N \le 500\,000$, $1 \le K \le N$, $1 \le d_i \le 10^9$.

Output

Output a single integer, representing the minimum possible maximum distance travelled by a dwarf when going to a neighbour and visiting a grocery store on the way, for optimal location of K grocery stores. It can be proven that the result is always an integer.

Examples

standard input	standard output
3 1 2 4 5	6
5 3 2 6 4 1 5	6

Problem H. Hues

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Dwarf the Painter, a master of magic colors, has N available buckets of paint, each containing a different color. Whenever a subset of these colors is applied on the same piece (of non-zero area) of a white canvas, it produces a particular hue. Different subsets of colors produce different hues.

Dwarf the Painter was challenged by the village elders to create a painting containing all possible $2^N - 1$ hues (this number excludes the hue of the canvas itself, where no color was placed). Being almost as lazy as Dwarf the Coach Potato, Dwarf the Painter painted N circles, each filled with one color. When he showed it to the elders, their jaws dropped in amazement at the sheer quality of the masterpiece. However, they were not at all sure whether all $2^N - 1$ hues were actually created in this way. Help them with this task.

Input

The first line of the input contains an integer T, the number of test cases.

The first line of each test case contains an integer N, representing the number of circles. The next N lines of each test case contain three integers x_i, y_i and r_i, separated by a single space, where (x_i, y_i) and r_i represent the coordinates of the center and radius of the circle filled with the i-th color. No three circles intersect at one point.

Limits $1 \le T \le 200$, $1 \le N \le 200$, $-1\,000 \le x_i, y_i \le 1\,000$, $1 \le r_i \le 1\,000$, the sum of N in all testcases does not exceed 200.

Output

The *description of a hue* is a sequence of N numbers, separated by a single space, where each number is equal either to 1 or to 0. The i-th color is used to create the hue if and only if the i-th number of the description is 1.

For each test case, if the painting contains all $2^N - 1$ hues, print one line with the word `YES`. Otherwise print two lines: the first one should contain the word `NO` and the second one should contain the description of one of the missing hues. If more than one hue is missing from the painting, your program can print the description of any of them. Remember that a hue is produced only if the corresponding colors are placed on a canvas piece of *non-zero* area. You may assume that all the circles in one test case are pairwise distinct.

Example

standard input	standard output
5 2 0 0 1 1 0 1 2 0 0 1 2 0 1 3 -1 0 2 1 0 2 0 1 2 5 0 0 4 5 -4 4 10 0 4 15 -4 4 20 0 4 5 0 0 7 0 3 4 3 0 4 0 -3 4 -3 0 4	YES NO 1 1 YES NO 1 0 1 0 0 NO 0 1 0 0 0

Problem I. Illuminati

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Dwarf the Illuminati is preparing the New Year's Eve light show. There will be N candles put in a straight line. Each candle can be either lit or not. The show will consist of K rounds. Between each pair of consecutive rounds, Illuminati will move the candles around.

To make everything easier to remember, Illuminati wants to move the candles in the same way each time. Formally, he will create an N-element permutation P, and after each round, he will move the candle from position i to position $P(i)$.

He already designed the entire show. Please help him and determine if there exists a permutation P that would match his requirements. If there are multiple such permutations, output the lexicographically smallest one.

Input

The first line contains two natural numbers: N and K. Each of the next K lines will contain a binary string of length N, where 1 means the candle is lit, and 0 means that it is not. The i-th of these lines corresponds to the i-th round of the show.

Limits $1 \leq N, K \leq 100\,000$, $N \cdot K \leq 1\,000\,000$.

Output

If the required permutation exists, in the first line output `YES`, and in the second line output the permutation (starting from 1). If such a permutation doesn't exist, output `NO`.

Examples

standard input	standard output
3 3 100 010 001	YES 2 3 1
4 2 0011 0110	YES 1 4 2 3

Problem J. Just Mining

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	1024 megabytes

The dwarfs have identified multiple new gold deposits at the bottom floor of their Deep Mine. The locations of the deposits (on the floor, i.e., plane) are given by a multiset of points $P = \{p_1, \ldots, p_m\}$, and each point has a corresponding (estimated) value, given by v_i. Mining these deposits is not so easy however, as too much drilling in certain areas might cause instability and risk collapsing of the whole mine. Together, the guilds of miners, geologists, and engineers, managed to identify these hazardous areas: They form a multiset of circles (on a plane) $C = \{c_1, \ldots, c_n\}$. Each circle c_i has an associated maximal capacity f_i, which is the maximum number of deposits contained within it that may be (safely) extracted.

Your task is to choose a subset of points $A \subseteq P$ maximizing the total (estimated) value of selected points such that no capacity is exceeded. Formally, for any $i \in \{1, \ldots, n\}$, let P_i be the multiset of points from P contained in the i-th circle (i.e., lying either strictly inside c_i or on its boundary). Then A should satisfy $|A \cap P_i| \leq f_i$ for all i.

Moreover it is guaranteed that C can be divided into two disjoint *laminar families* C_1, C_2. A *laminar family* of circles is defined here as a set of circles in which each pair of circles is either disjoint or contained in one another. That means that no two circles in one laminar family have a point in common.

Input

The first line of the input contains two integers N and M, the number of circles and points, respectively.

Each of the next N lines describes one of the circles. Such description contains four integers x_i, y_i, r_i, f_i describing in order: coordinates of the center of i-th circle, its radius, and capacity.

Each of the next M lines describes one of the points. Such description contains three integers x_i, y_i, v_i describing in order: coordinates of the point and its value. Multiple points may have the same coordinates.

Limits $1 \leq N, M \leq 300$, $-10^9 \leq x_i, y_i \leq 10^9$, $1 \leq r_i, v_i \leq 10^9$, $1 \leq f_i \leq 300$.

Output

As an output print three lines.

The first line should contain one integer – the maximum possible total value of points in multiset A.

The second line also should contain one integer – the number of points in a multiset A obtaining such value.

The third and final line should contain numbers of points in such a multiset. The points are numbered according to their order in the input, starting from 1. The points can be given in any order, but each of them must appear at most once. If there are many such sets you can print any of them.

Examples

standard input	standard output
5 5 3 5 2 2 4 10 4 2 5 10 2 3 5 5 5 2 14 4 2 3 6 11 3 3 8 5 4 6 20 9 5 4 14 5 1	28 4 1 3 4 5
3 2 4 7 2 2 4 8 1 1 8 7 1 1 4 7 5 4 8 4	5 1 1

Problem K. King's Festival

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	1024 megabytes

For the upcoming festival, the Dwarf King wants to light up the town hall of the kingdom. The hall is a grid consisting of $N \times N$ square tiles, and all these tiles must be illuminated. However, in order to save energy and preserve the magical power of the lanterns, the king wants to use as few lanterns as possible. He decided to limit the placement of these lanterns to the main diagonal of the grid, which runs from the top-left corner to the bottom-right corner.

The magic lanterns were crafted a long time ago and have a unique power: not only they illuminate the tile on which they stand, but they also cast long beams of light in specific patterns. That is, when placed on a floor tile, each lantern would light up:

- all the tiles in its row,
- all the tiles in its column, and
- all tiles in both diagonals passing through its tile.

Some of these lanterns have already been placed by the dwarf engineers, fixed firmly to the ground, and they could not be moved.

Input

The input is just one line describing the lanterns already placed on the main diagonal. It contains a single string where each character is either `.` or `#`. The i-th square of the diagonal already contains a lantern if and only if the i-th character is equal to `#`.

Limits String from the input contains at least 1 and at most 64 characters.

Output

The first line of the output should contain a single integer: the smallest number of lanterns that need to be placed on the main diagonal to illuminate all the squares of the town hall (including those already placed). The second line of the output should contain the description of the lantern placement in the same format as in the input; this should include the lamps placed initially. If there are multiple correct solutions, you may output any of them.

Examples

standard input	standard output
`......`	`4` `###.#.`
`.#.#...`	`4` `.###.#.`

Problem L. Labyrinth

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Dwarf the Puzzler is designing labyrinth puzzles for other dwarfs. A labyrinth is an $N \times N$ square with each cell being either a passage or a wall. In one of the passages is located a pawn, which faces one of four possible directions: right, up, left or down. In each step a dwarf first rotates the pawn and then moves it in the resulting direction. Of course the pawn cannot move to or through the wall! If the pawn moves out of the labyrinth, then the puzzle is solved.

Dwarfs love those puzzles, but Dwarf the Riddler, Puzzler's old rival, boasts that labyrinths are dull and that he can solve them using a simple tactic: *among the cells to the right, front, left and back (relatively to the direction faced by the pawn) choose the first one that is a passage, then rotate the pawn to this direction and move in this direction.* The Puzzler is not convinced that this indeed is a proper tactic and, as an engineer, wants to run some tests on it. Help him, for the labyrinths he devised and starting positions (and directions) of the pawn compute, whether the pawn will eventually leave the labyrinth and how many moves it will take.

Input

The first line of input contains two integers N and Q, representing the size of the labyrinth and the number of queries, respectively.

In the next N lines, there is a description of the labyrinth, with description of one labyrinth row per line, starting from top-most row. Each line contains N characters that are either a dot `.` or a hash `#`, representing a passage and wall, respectively. Rows and columns of the labyrinth are numbered from 1 to N, top-to-bottom and left-to-right.

The following Q lines contain descriptions of the subsequent queries. Each query consists of two integers r, c, and one character d. The numbers r and c represent the row and column where the pawn is initially located, and the character d indicates the direction the pawn is facing, where the letters `U`, `D`, `L` and `R` mean that the pawn is initially facing respectively up, down, left or right.

You can assume that the pawn is never located in a cell where there is a wall.

Limits $1 \le N \le 1\,000$, $1 \le Q \le 100\,000$, $1 \le r, c \le N$, $d \in \{\texttt{U}, \texttt{D}, \texttt{L}, \texttt{R}\}$.

Output

For each query, output one line containing a single integer: the number of moves in which the pawn will leave the labyrinth. If the pawn never leaves the labyrinth, output -1 instead.

Example

standard input	standard output
10 10 ########## #.#.#.#... #.....#.## ####.##..# #....##### #..#.#...# #.#..#.#.# #.#.##.#.# #.#......# ###.###### 2 10 U 2 10 L 4 8 U 10 4 U 8 7 U 8 9 U 6 2 U 5 2 U 3 5 D 7 4 R	1 11 7 61 -1 54 11 -1 33 4