Problem A. ABBA

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

In this problem, we operate with tables of fixed size $h \times w$ consisting of real values. Let's define an addition operation on two tables as their component-wise sum.

A *multiplication table* for two real vectors $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_h)$ and $\beta = (\beta_1, \beta_2 \ldots, \beta_w)$ is the table $T_{\alpha,\beta}$ where the element at the intersection of i-th row and j-th column is $\alpha_i \cdot \beta_j$.

You start with a table of size $h \times w$ consisting of zeroes. In one turn, you are allowed to add a multiplication table for two arbitrary real vectors α of length h and β of length w to the current table. Your task is to make the current table equal to a goal table G in the minimum number of turns. What is the minimum number of turns you have to perform?

Input

The first line of input contains two integers h and w ($1 \le h, w \le 200$).

The i-th of the following h lines contain w space-separated **integers** $a_{i,1}$, $a_{i,2}$, ..., $a_{i,w}$ ($-10^6 \le a_{i,j} \le 10^6$), where $a_{i,j}$ is the value on the intersection of i-th row and j-th column of the goal table G.

Output

If it's impossible to obtain the goal table G, print "`-1`" (without the quotes). Otherwise, output the minimum number of turns you have to perform in order to achieve it.

Examples

standard input	standard output
3 5 1 2 3 4 5 2 4 6 8 10 3 6 9 12 15	1
3 3 2 0 2 0 2 0 2 0 2	2

Note

In the first sample, the table T can be obtained using $\alpha = \begin{pmatrix}1 & 2 & 3\end{pmatrix}$, $\beta = \begin{pmatrix}1 & 2 & 3 & 4 & 5\end{pmatrix}$.

In the second sample, the table T can be obtained as sum of $T_{\alpha_1,\beta_1} = \begin{pmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{pmatrix}$ for vectors $\alpha_1 = \begin{pmatrix}1 & 1 & 1\end{pmatrix}$, $\beta_1 = \begin{pmatrix}1 & 1 & 1\end{pmatrix}$ and $T_{\alpha_2,\beta_2} = \begin{pmatrix}1 & -1 & 1\\ -1 & 1 & -1\\ 1 & -1 & 1\end{pmatrix}$ for vectors $\alpha_2 = \begin{pmatrix}-1 & 1 & -1\end{pmatrix}$, $\beta_2 = \begin{pmatrix}-1 & 1 & -1\end{pmatrix}$.

Problem B. Black Sabbath

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

Consider a two-dimensional rectangular pie of size $w \times h$. Let's introduce a coordinate system such that corners of the pie are points $(0, 0)$, $(w, 0)$, (w, h), and $(0, h)$. The pie contains n cherries. Each cherry is represented by a circle of radius r_i lying strictly inside the pie. No two cherries overlap or touch each other or the border of the pie.

You invited $n - 1$ guests to a party, so there are n people including you. Now it's time to cut the pie and eat it! A person is satisfied if he or she gets a strictly convex polygonal piece of the pie containing a cherry strictly inside. The size and form of the piece don't matter, as well as the size of the cherry. A polygon is called strictly convex if, for each two points a and b inside it, it also contains the whole segment $[a, b]$ between those two points, and additionally, no three vertices of the polygon lie on a same line.

You spent lots of hours cooking this beautiful pie, so you don't want to leave any unused pieces of that wonderful dish. So, you want to cut the pie into exactly n pieces according to the rules above without leaving any extra pieces. Your task is to find any such cutting.

Input

The first line of input contains three integers w, h and n ($4 \le w, h \le 10^4$, $1 \le n \le 1000$), the size of the pie and the number of cherries.

The following n lines contain triples of integers x_i, y_i, r_i ($1 \le r_i \le \frac{\min\{w,h\}}{2} - 1$, $r_i + 1 \le x_i \le w - r_i - 1$, $r_i + 1 \le y_i \le h - r_i - 1$), the coordinates of the center of i-th cherry and its radius.

It is guaranteed that the distance between any two distinct cherries is at least 0.1.

Output

Output descriptions of n pieces containing each cherry in the order they follow in the input. Each description must start with an integer k ($k \ge 3$), the number of sides of a strictly convex polygon that forms the corresponding piece. After that, output k lines containing coordinates of polygon's vertices in counter-clockwise order.

It is recommended to output all floating-point numbers with no less than 9 digits after the decimal point. The checking program will perform at least the following checks using 80-bit floating-point arithmetic:

- Each side of each piece must have length at least 10^{-4}.
- The internal angle between any two consecutive sides of any piece must be no larger than $\pi - 10^{-11}$ radians. The vertices of each piece must be given in counter-clockwise order.
- The total area of all pieces must be in the range $[(1 - 10^{-9}) \cdot w \cdot h, (1 + 10^{-9}) \cdot w \cdot h]$.
- The distance between each point of a cherry and the border of the piece containing that cherry must be at least 10^{-4}.

- A point of one piece can belong to another piece as well, but in such case, the distance from this point to the border of the second piece can be at most 10^{-4}.
- A point of a piece can lie outside the pie, but in such case, the distance from this point to the border of the pie can be at most 10^{-4}.
- The total number of sides of all pieces can be at most $2 \cdot 10^4$.

As you can see, writing a checking program can also be a hard technical problem for a programming contest.

Example

standard input	standard output
12 10 5 3 2 1 9 2 1 6 5 2 3 8 1 9 8 1	5 0.0 0.0 6.0 0.0 6.0 1.0 1.5 5.0 0.0 5.0 5 6.0 0.0 12.0 0.0 12.0 5.0 10.5 5.0 6.0 1.0 4 6.0 1.0 10.5 5.0 6.0 9.0 1.5 5.0 5 0.0 5.0 1.5 5.0 6.0 9.0 6.0 10.0 0.0 10.0 5 12.0 10.0 6.0 10.0 6.0 9.0 10.5 5.0 12.0 5.0

Note

If you eventually solve all problems of the contest and you feel lonely and bored, ask the authors to give you a version of this task with $n \leq 10^5$:) .

Problem C. Mr. Credo

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

Consider the two-dimensional plane Oxy. There are n spotlights. When the i-th spotlight is placed at some point and turned on, it lights up all the points belonging to the interior or the border of an angle of measure ϕ_i ($0^\circ < \phi_i < 360^\circ$) and vertex in the chosen point (its vertex is also considered to be lit up). Each spotlight can be rotated arbitrarily and may be placed in an arbitrary point of the plane.

You and your friends want to light up the whole plane in order to protect something very important from the intrusion. First you tried to put all spotlights into the same point, but after some attempts you discovered that there exists no way to rotate all the spotlights so that they light up every point on the plane. After that, one of your friends suggested that, if you spread the spotlights in some optimal way, then it would be possible to light up the whole plane. He even told the points where you should put the spotlights, as well as how you should rotate them.

Having an enormous mathematical intuition, you suspect that he is wrong, and the whole plane still isnt lit up. To prove that he is wrong, you are even going to present an uncovered circle of radius 1 (that is, all points belonging to the interior of that circle and its border shouldnt be lit up), that denotes an intruder that is able to hide in the darkness, not being lit by your spotlights.

Write a program which, given the description of all spotlights, finds a point (x_0, y_0) such that the circle of radius 1 with center in (x_0, y_0) is completely uncovered by the spotlights.

Input

The first line of input contains a single integer n ($1 \le n \le 2 \cdot 10^5$), the number of spotlights.

The following n lines contain descriptions of spotlights. Each spotlight description consists of four integers x_i, y_i, ϕ_i, α_i ($-50 \le x_i, y_i \le 50$, $0 < \phi_i < 1\,296\,000$, $0 \le \alpha_i < 1\,296\,000$). Here, x_i and y_i are coordinates of the corresponding spotlight, ϕ_i and α_i are angles measured in arcseconds (see Notes section for clarification), equal to the angular measure of the spotlight and the angle of its rotation respectively.

The definition of a rotation angle is the following. Consider the horizontal ray in positive direction of x-coordinate with the initial point at (x_i, y_i), denote it as l_0. Let's denote as l_γ the result of rotation of l_0 around the point (x_i, y_i) in counter-clockwise direction by angle γ. Then the spotlight lights up all points that are sweeped when a ray is being rotated from direction l_{α_i} to direction $l_{\alpha_i+\phi_i}$ in counter-clockwise direction.

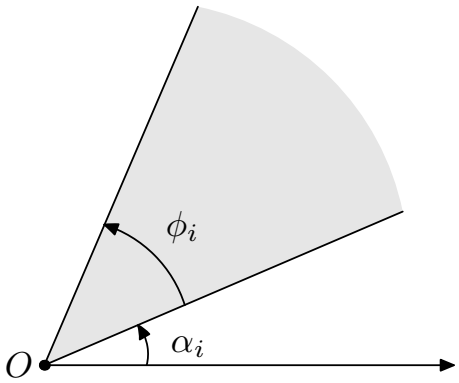

Illustration for the input format.

It is guaranteed that the given set of spotlights can't light the whole plane if they are all put into a single point and rotated arbitrarily.

Output

If your intuition failed, and the whole plane is being actually lit up, print the only word "NO".

Otherwise, on the first line, print the only word "YES", and on the following line, print two **integers** x_0, y_0 satisfying the condition $-10^9 \le x_0, y_0 \le 10^9$, the coordinates of the center of some uncovered circle. It is guaranteed that if there exists an uncovered point, then there also exists an uncovered circle satisfying the conditions above.

Example

standard input	standard output
4 1 2 486000 0 -1 1 324000 648000 1 0 108000 0 1 0 108000 1188000	YES -3 3

Note

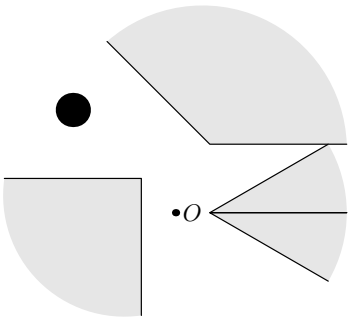

Illustration for the sample test.

An arcsecond is a unit of angular measurement, it is denoted as $''$. It is equal to $\frac{1}{3600}$ of a degree, that is, the full circle is equal to $1\,296\,000''$.

Problem D. Deep Purple

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 mebibytes

It's always so cool to generalize well-known algorithms a bit so that they become less trivial!

Your are given a string S. Your task is to process q so-called π-queries. Each π-query is determined by two integer parameters l and r ($1 \le l \le r \le |S|$). The answer for a π-query is the largest non-negative value $x \le r - l$ such that $S[l \dots l + x - 1] = S[r - x + 1 \dots r]$ (all ranges are inclusive, all indices are 1-based). Note that $x = 0$ always satisfies the given condition because both parts of the equation are empty strings.

For example, the result of a π-query for string $S =$ "`gabacababad`", $l = 2$ and $r = 8$ is 3, since $S[2..4] = S[6..8] =$ "`aba`", and no larger value satisfies the condition above.

Input

The first line of input contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$), the length of the string S and the number of queries.

The second line contains the string S consisting of n lowercase English letters.

Each of the next q lines contain two positive integers l_i, r_i ($1 \le l_i \le r_i \le n$) that describe the i-th π-query.

Output

Print answers for each of the q queries keeping the order from the input.

Example

standard input	standard output
11 3 gabacababad 2 8 1 3 6 10	3 0 3

Problem E. Elvis Presley

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

Consider a possibly infinite directed graph $G = (V, E)$ without loops (that is, for each $(u, v) \in E$, $u \neq v$). Let's define its transitive closure $G^+ = (V, E^+)$ as the directed graph having the same set of vertices V such that its edges are all pairs (u, v) for which $u \neq v$ and there exists a finite path $u = p_0, p_1, p_2, \ldots, p_{k-1}, p_k = v$ such that $(p_{i-1}, p_i) \in E$ for all $i = 1, 2, \ldots, k$.

An antichain is a (possibly infinite) set $A \subseteq V$ such that, for any two distinct vertices $u, v \in A$, none of the two edges (u, v) and (v, u) are present in G^+.

Consider a set U and some property P defined for all its subsets (an example of a property defined for each subset $A \subseteq V$ is whether A is an antichain of the graph $G = (V, E)$). We say that subset $S \subseteq U$ is a *maximum* subset satisfying property P if its size $|S|$ (that is possibly equal to ∞) is maximum possible. We say that $S \subseteq U$ is a *maximal* subset satisfying property P if there exists no other subset T satisfying P such that $S \subsetneq T$, that is, S can't be extended to become a larger subset satisfying the same property. Note that those two definitions are different: if a maximum subset satisfying P exists and it is finite, it is obviously also maximal, but a maximal subset satisfying P may not be maximum.

We define *minimal* and *minimum* subsets satisfying some property P in a similar manner.

Let's define an EP-graph: $EP = (V_{EP}, E_{EP})$ such that its vertices are $V_{EP} = \mathbb{N} = \{1, 2, 3, \ldots\}$ and its edges are $E_{EP} = \{(v, 2v) : v \in \mathbb{N}\} \cup \{(v, 2v + 1) : v \in \mathbb{N}\}$.

You are given two distinct vertices a and b of V. Find a *minimum maximal* antichain in EP containing both a and b, or determine that there are no such antichains. If there are several possible answers, find any of them.

More formally:

$AC = \{A \subseteq V_{EP} : A \text{ is an antichain in } EP\}$.

$MaxAC = \{A \in AC : A \text{ is maximal}\}$.

$MinMaxAC = \{A \in MaxAC : |A| = \min\limits_{S \in MaxAC} |S|\}$.

Your task is to find any element of $MinMaxAC$.

Input

The first line of input contains two integers a and b ($1 \leq a, b \leq 10^9$, $a \neq b$).

Output

If there exists no minimum maximal antichain in EP containing both a and b, or if it is infinite, print -1. Otherwise, print elements of any minimum maximal antichain in EP containing both a and b in ascending order.

Examples

standard input	standard output
2 7	2 6 7
1 2	-1

Problem F. Frank Sinatra

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	256 mebibytes

You are given a bidirectional graph T which is a tree consisting of n vertices and $n - 1$ edges. Each edge of the tree is associated with some non-negative integer x_i.

Your task has a very simple description. You are given q queries. In j-th, query you have to find the smallest non-negative integer y that is not present in the set of all integers associated with edges of the simple path between vertices a_j and b_j.

Input

The first line of input contains two integers n and q ($2 \le n \le 10^5$, $1 \le q \le 10^5$), the number of vertices of the tree and the number of queries.

The following $n-1$ lines contain triples of integers u_i, v_i, x_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le x_i \le 10^9$), each denoting an edge (u_i, v_i) associated with an integer x_i.

The following q lines contain pairs of integers a_j, b_j ($1 \le a_j, b_j \le n$), each denoting a query about the path between vertices a_j and b_j.

Output

For each query, output one line containing the smallest non-negative y such that there is no edge associated with y lying on the corresponding simple path.

Example

standard input	standard output
7 6 2 1 1 3 1 2 1 4 0 4 5 1 5 6 3 5 7 4 1 3 4 1 2 4 2 5 3 5 3 7	0 1 2 2 3 3

Problem G. Green Day

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

Consider a graph consisting of $n \geq 2$ vertices without loops or parallel edges. Each edge may be colored in one of k colors. We call a coloring *proper* if edges of each color form a spanning tree of the graph (that is, for each color c, there exists a unique path between each pair of vertices that uses only edges of color c). Denote such spanning tree for color c as T_c.

We call a *proper* coloring *safe* if for each two colors c and d and for each two distinct vertices u and v, the following statement is correct: $\text{path}_{T_c}(u, v) \cap \text{path}_{T_d}(u, v) = \{u, v\}$, where $\text{path}_T(u, v)$ is the set of all vertices of tree T that lie on the simple path between u and v (including u and v themselves).

Your task is to construct such a graph that its edges are colored in k colors forming a *safe proper* coloring.

Input

The first and only line of input contains a single positive integer k ($2 \leq k \leq 100$), the number of colors you should use in your graph.

Output

On the first line, output $n \geq 2$: the number of vertices in your graph.

Then, output k groups consisting of $n - 1$ edges representing edges of each color. Output each edge as a pair of integers a_i, b_i on a separate line ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$).

Your output must satisfy the condition $(n - 1) \cdot k \leq 10^6$. There must be no parallel edges.

You are allowed to output any valid answer. It's guaranteed that at least one solution exists.

Example

standard input	standard output
2	4 1 2 1 3 3 4 4 1 2 3 2 4

Problem H. Hans Zimmer

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

Hans wants to become a glass carver (a person who creates beautiful artwork by cutting the glass). He already has a rectangular piece of glass of size $w \times h$ millimeters, a diamond glass cutter and lots of enthusiasm. What he lacks is understanding of what to carve and how.

In order not to waste time, he decided to practice the technique of carving. To do this, he makes vertical and horizontal cuts through the entire sheet. This process results in making smaller rectangular fragments of glass. Hans does not move the newly made glass fragments. In particular, a cut divides each fragment of glass that it goes through into smaller fragments.

Hans doesn't know how to make a great artwork, so he performs random cuts as follows. First, he tosses a fair coin to determine if he is going to cut the glass vertically or horizontally (that is, the probability of choosing each direction is 50%). After that, he chooses a uniformly distributed random real point on the corresponding side of the rectangle, and makes a cut through that point. All n random points and all n coin tosses are mutually independent.

Hans is going to perform exactly n cuts. What he is interested in, is the fragment with the smallest area that is formed after he makes all cuts. Denote its area as ξ. Your task is to calculate $\mathrm{E}[\xi]$, the expected value of ξ.

Input

The only line of input contains three space-separated integers w, h and n ($1 \le w, h \le 10^3$, $1 \le n \le 10^6$), the size of the piece of glass and the number of cuts Hans is going to perform.

Output

Output the expected area of the smallest fragment formed after performing all cuts. Your answer will be considered correct if its **relative** error is no more than 10^{-4} (note that having absolute error no more that 10^{-4} is **not enough**).

Examples

standard input	standard output
2 4 1	2
42 24 2	87.5

Problem I. Ivan Dorn

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

You are given a sequence consisting of n integers $a_1, a_2, \ldots, a_n$. Let's call some contiguous segment of this sequence $a_l, a_{l+1}, \ldots, a_{r-1}, a_r$ a *canyon* if $a_l = a_r$ and for each integer $l \le x \le r$, the inequality $a_x \le a_l$ holds. In particular, $l = r$ automatically means that the segment is a canyon. The length of a canyon is considered to be equal to $r - l$.

Your task is to answer m queries of the following form: for a given contiguous segment $a_l, a_{l+1}, \ldots, a_{r-1}, a_r$ defined by its endpoints l and r, find a canyon of maximum length that is a subsegment of this segment.

Input

The first line of input contains two integers n and m ($1 \le n, m \le 5 \cdot 10^5$), the length of the sequence and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$).

Each of the following m lines contains two positive integers l_i and r_i which describe the queries ($1 \le l_i \le r_i \le n$).

Output

For each of the m queries, print the maximum length of a canyon inside the given segment on a separate line.

Example

standard input	standard output
8 5 4 3 2 2 3 3 7 3 1 7 6 8 1 3 3 6 1 8	4 0 0 1 4

Note

In the sample test, the possible maximal canyons for each of the queries are: $(2, 6)$, $(6, 8)$, $(1, 1)$, $(3, 4)$ and $(2, 6)$.

Problem J. Jimi Hendrix

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

You are given a tree T consisting of n vertices and $n-1$ edges. Each edge of the tree is associated with a lowercase English letter c_i.

You are given a string s consisting of lowercase English letters. Your task is to find a simple path in the tree such that the string formed by concatenation of letters associated with edges of this path contains string s as a subsequence, or determine that there exists no such simple path.

Input

The first line of input contains two positive integers n and m ($2 \le n \le 5 \cdot 10^5$, $1 \le m \le n-1$), the number of vertices in the tree and the length of the string s.

The following $n-1$ lines contain triples u_i, v_i, c_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, c_i is a lowercase English letter), denoting an edge (u_i, v_i) associated with letter c_i.

The last line contains a string s ($|s| = m$) consisting of lowercase English letters.

Output

If the desired path exists, output its endpoints a and b. Otherwise, output "`-1 -1`". If there are several possible answers, you are allowed to output any of them.

Example

standard input	standard output
9 3 1 2 a 2 3 b 2 4 a 4 5 b 4 6 c 6 7 d 6 8 a 8 9 b acb	8 3

Problem K. Korn

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

Consider a connected graph $G = (V, E)$ without loops and parallel edges. Let's define a *complete walk* starting in vertex v as a sequence of visited vertices $v = p_0, p_1, p_2, \ldots, p_k = u$ such that the following conditions are satisfied:

1. For each $i = 1, 2, \ldots, k$, the unordered pair $(p_i, p_{i-1}) \in E$, that is, each two consecutive vertices are connected by an edge.
2. Each edge (a, b) appears at most once among all (p_i, p_{i-1}), that is, the walk p doesn't pass through the same edge twice.
3. There exists no vertex p_{k+1} that can be appended at the end of the walk such that the previous two conditions are still satisfied.

The vertex u is called the *terminal* vertex.

The vertex v is called *unavoidable* if any complete walk starting in v visits all edges in the graph (that is, $k = |E|$), and its terminal vertex is also v.

Your task is to find all *unavoidable* vertices in a given graph.

Input

The first line of input contains two integers n and m ($3 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 5 \cdot 10^5$), the number of vertices and the number of edges in the graph respectively.

The following m lines contain pairs of integers a_i, b_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$), denoting endpoints of i-th edge.

It is guaranteed that the graph contains no loops and no parallel edges, and also that it is connected.

Output

Print the number of *unavoidable* vertices on the first line of output, and 1-based indices of all *unavoidable* vertices on the second line in ascending order.

Example

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

Note

In the sample, for example, vertex 4 is not unavoidable because there exists a complete walk 4, 5, 2, 1, 4 that terminates in 4 but that doesn't visit all edges in the graph.