Problem A. CNF-SAT

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	512 mebibytes

The annual $P = NP$ *Equality Parade* is just around the corner. Bytiana – the leader of the $P = NP$ movement, decided to silence his opponents by announcing a proof of the famous equality.

Bytiana proves $P = NP$ by showing a polynomial time algorithm for a well known NP-complete problem called *CNF-SAT*. In this problem we are given n boolean variables $x_1, \ldots, x_n$ and a boolean formula in the *conjunctive normal form*. Such a formula is of the form

$$(l_{1,1} \vee \ldots \vee l_{1,q_1}) \wedge (l_{2,1} \vee \ldots \vee l_{2,q_2}) \wedge \ldots \wedge (l_{m,1} \vee \ldots \vee l_{m,q_m}),$$

where every expression $(l_{i,1} \vee \ldots \vee l_{i,q_i})$ is called a *clause* and every expression $l_{i,j}$ is a *literal*, i.e., one of the variables $x_1, \ldots, x_n$ or its negation. We assume that no correct clause contains two identical literals. For example, $(x_1 \vee \neg x_3) \wedge (x_2) \wedge (\neg x_3 \vee \neg x_1 \vee x_2)$ is a formula in conjunctive normal form for $n = m = 3$.

The CNF-SAT problem asks whether there exists an interpretation of variables $x_1, \ldots, x_n$ so that the given formula is satisfied (i.e., it evaluates to true).

Unfortunately, Bytiana's proof lacks just one step. She claims that he managed to reduce[1] the general CNF-SAT problem to its special case where every clause C is *contiguous*, i.e., it has the following properties:

- For every i, x_i and $\neg x_i$ cannot simultaneously be literals of C.
- If i, j, k satisfy $i < j < k$ and C contains both the variable x_i (or its negation) and x_k (or $\neg x_k$) then it also contains x_j (or $\neg x_j$).

For example, for $n = 3$, the clauses (x_2) and $(\neg x_3 \vee \neg x_1 \vee x_2)$ are contiguous; on the other hand, $(x_2 \vee \neg x_2)$ and $(x_1 \vee \neg x_3)$ are not contiguous.

Help Bytiana and find an effective algorithm solving this special case of CNF-SAT. To surprise him even more, write a program that counts the number of interpretations of variables $x_1, \ldots, x_n$ satisfying the given CNF-SAT formula consisting of contiguous clauses only.

Input

The first line of the input contains an integer n ($1 \le n \le 1\,000\,000$) denoting the number of variables. The second line contains a description of a CNF-SAT formula using variables $x_1, \ldots, x_n$ (some of them may be absent, though) consisting of contiguous clauses only. The formula is given in the following format (take a look at the sample for clarity).

- Each clause begins with an opening parenthesis `(` and ends with a closing parenthesis `)`.
- A literal x_i (for $1 \le i \le n$) is represented as `x`i and a literal $\neg x_i$ is represented as `~x`i, e.g., `x2` or `~x15`.
- Consecutive literals within one clause are separated with a character `v` (denoting the logical *or*) surrounded by single spaces.
- Consecutive clauses are separated with a character `^` (denoting the logical *and*) surrounded by single spaces.

The total number of literals in all clauses of the input formula is not be greater than 1 000 000.

[1] Bytiana forgot to mention whether her reduction works in polynomial time...

Output

You should output the number of interpretations of variables $x_1, \ldots, x_n$ satisfying the given formula, modulo $10^9 + 7$.

Example

standard input	standard output
3 (x2) ^ (x3 v ~x2) ^ (x2 v x1 v ~x3)	2

Note

Explanation for example: the input formula is satisfied for only two interpretations: $(0, 1, 1)$ and $(1, 1, 1)$.

Problem B. Almost pattern matching

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

We say that a string s is *almost equal* to a string r if and only if s can be transformed into r using at most k operations, each of one of the following types:

- an insertion of a letter at the beginning of s, at the end of s, or between its two already existing letters,
- a removal of a single letter of s,
- a replacement of a single letter of s with a different letter.

You are given two strings t and p. You need to determine whether t contains a substring[2] almost equal to p.

Input

In the first line of the input there is an integer k ($0 \le k \le 10$) used to define the relation of being almost equal. In the second and the third lines, respectively, there are strings t and p. Both strings consist of at least one and at most 100 000 lowercase Latin letters.

Output

If the string t does not contain any substring that is almost equal to p, you should output `NO`.

Otherwise in the print two integers a, b ($1 \le a \le b \le |t|$) such as substring t starting at the position a and ending and position b is almost equal to p. If here are more than one solutions, print any.

Example

standard input	standard output
1 abcd abd	1 2
1 abcd bde	NO

[2]For a given string $t = c_1 c_2 \cdots c_n$, a *substring* of t is any of the strings of the form $c_a \cdots c_b$, where $1 \le a \le b \le n$.

Problem C. Two paths

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

In Byteland, the cities are connected by one-way roads. The road structure is peculiar: if you leave some city using one of the roads, you can never return to this city following these roads. To put it another way: the road structure can be described as a directed acyclic graph.

The way the roads have been built created some problems. For example, some cities might be unreachable even from the Byteland's capital city. Even worse – the roads are often closed and renovated; in such a situation, we might be unable to reach some cities even when it was possible to reach them before.

Byteasar lives in the capital city and often travels to other Byteland's cities. For each city C, he would like to know if there are two travel routes from the capital to C which don't share any roads. If this is the case (or if C is the capital city), he knows that his travel to C is always possible even if one of the roads is renovated.

Help Byteasar and find all the cities which can be reached from the capital city even if some road is closed.

Input

The first line of the input contains two integers n, m ($1 \le n \le 200\,000$, $1 \le m \le 500\,000$) – the number of cities in Byteland and the number of roads connecting them, respectively. The cities have labels $1, 2, \ldots, n$ and the capital city is assigned the label 1.

The following m lines contain the road structure description; the i-th of these lines contains two integers u_i, v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) denoting that the i-th one-way road starts in the city u_i and ends in the city v_i.

Output

The first line of the output should contain the number of cities which can be reached from the capital city even if one of the roads is closed.

The following line should contain the labels of these cities, sorted increasingly. Print single spaces between the labels.

Example

standard input	standard output
7 9 1 2 1 3 3 4 4 5 2 4 2 5 5 6 5 7 5 7	4 1 4 5 7

Problem D. Mushrooms after rain strike back

Input file:	*standard input*
Output file:	*standard output*
Time limit:	8 seconds
Memory limit:	512 mebibytes

The rain is pouring again over Bytean Forest. As everybody knows, mushrooms grow really fast after rain. Byteasar is a passionate mushroom picker. Thus, he cannot miss such a great opportunity! He plans to collect as many mushrooms as possible. He took with him a drone that is able to fly to a clearing and collect all mushrooms from that clearing. It takes exactly one day to perform such an action.

Bytean Forest is known for its unique kind of mushrooms – *Agaricus linearis*. Its name originates from the fact that for a particular clearing there is a fixed amount of newly grown mushrooms every day.

There are n clearings in the forest. On the first day there are b_i mushrooms on the i-th clearing and there are a_i newly grown mushrooms every day. Byteasar will collect mushrooms for k consecutive days starting from the first one. Every day he will send his drone to some clearing (possibly one that his drone already collected mushrooms from) to collect the mushrooms. Help Byteasar determine the maximum number of mushrooms he can collect this way. Byteasar has not decided yet how many days he will spend on mushroom picking. Your task is to determine the biggest number of collected mushrooms for every k from the interval $[1, n]$.

Input

In the first line of the input there is an integer n ($1 \le n \le 1\,000\,000$), denoting the number of clearings in Bytean Forest. The following n lines contain descriptions of clearings. The i-th of them contains two integers a_i, b_i ($0 \le a_i \le 1\,000\,000$, $0 \le b_i \le 10^{12}$), denoting the number of new mushrooms growing each night and the initial number of mushrooms on the i-th clearing, respectively.

Output

You should output n lines. The k-th of them should contain the maximum number of mushrooms that can be collected by Byteasar when using the drone in an optimal way and if the mushroom picking lasts for exactly k days.

Example

standard input	standard output
3 5 10 16 0 5 10	10 26 57

Problem E. Epic battle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Alice and Bobby are world-class masters in a popular card game *Magical Creatures*. Tonight, an epic battle between them will finally decide who will obtain a glorious title of the Grandmaster.

The rules of *Magical Creatures* are pretty original. Each of the players is in possession of n distinct decks of cards. The main game is preceded by a phase of choosing the decks. Starting with Alice, the players alternately discard the opponent's decks, one by one. Once each of the players is left with a single deck, the more exciting, final phase of the game begins. What's interesting, the gameplay in the final phase does not depend on the players' moves and even on their luck. For each pair of decks, we can tell beforehand how the game will end – either in Alice's win, Bobby's win or a draw. The winner of the final phase becomes the Grandmaster.

Your task is to check if Alice has a winning strategy. We say that Alice has a winning strategy if she can win (and become the Grandmaster) independently of Bobby's moves in the first phase of the game. If Alice cannot win when Bobby plays optimally, you should check whether she is at least able to draw, that is, prevent Bobby from receiving the title of Grandmaster.

Input

The first line of the input contains a single integer t ($1 \le t \le 20$) – the number of testcases. Then t descriptions of the testcases follow.

The first line of a single testcase contains two integers n, m ($1 \le n \le 100\,000$, $0 \le m \le 200\,000$) – the number of each player's decks and the number of pairs of decks, whose choice in the final phase does not end in a draw. Each player's decks are numbered from 1 to n.

Each of the following m lines contains three tokens a w b ($1 \le a, b \le n$, $w \in \{<, >\}$). If $w = <$, then the a-th Alice's deck loses with the b-th Bobby's deck. If $w = >$, then the a-th Alice's deck wins with the b-th Bobby's deck. For each ordered pair of integers (a, b), a line a w b (for any w) occurs at most once. If it does not occur at all, then the a-th Alice's deck draws with the b-th Bobby's deck.

Output

You should output t lines, one for each testcase, in the order they were given in the input. For each case, output:

- `WIN`, if Alice can win;
- `DRAW`, if Alice cannot guarantee a win, but she can avoid losing;
- `LOSS` otherwise.

Example

standard input	standard output
3 5 5 5 > 5 1 > 5 3 > 5 4 > 5 2 > 5 2 2 1 > 1 1 > 2 1 1 1 < 1	WIN DRAW LOSS

Problem F. Reachability

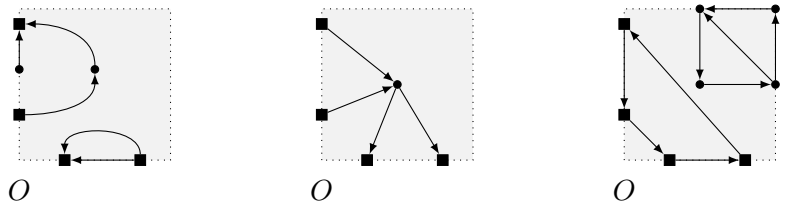
Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	512 mebibytes

Byteotia is an island located in the middle of an ocean and is rich in valuable natural resources. Byteotia's area is square and its four shores are facing four cardinal directions. A number of harbours have been built on its western and southern shores. So far there have been no roads connecting the harbours of the island. Unfavorable ocean currents made sailing between the western and the southern shores very difficult. As a result, it has been decided that Byteotia needs to develop a road network connecting the harbours. Unfortunately, there is too little money to build complex structures like tunnels, flyovers or overpasses.

For simplicity, let us assume that Byteotia is a square with the opposite corners located at points $(0, 0)$ and $(10^9, 10^9)$ of a Cartesian coordinate system. The n harbours on the western shore are located on the OY axis and the m harbours on the southern shore are located on the OX axis. The harbours' locations are distinct.

The plan is to construct some finite number of junctions and *one-way* roads connecting junctions and/or harbours. The junctions and the roads are collectively called *a road network*. For any road network, the harbours and the junctions have to be placed at distinct locations. The roads can be arbitrary curves without self-intersections, fully contained within the island's area, and with endpoints in either the junctions or the harbours. The roads cannot intersect at any points except their endpoints.

The below picture shows three example road networks for $n = m = 2$. The grey area denotes the island of Byteotia; the black squares depict the harbours and the black disks denote the constructed junctions.



Obviously, there are infinitely many possible roads networks. We say that two roads networks A and B are equivalent if and only if for every harbour x on the western shore and for every harbour y on the southern shore of the island, we can reach y from x using the network A if and only if we can reach y from x using the network B. In the above example the networks in the middle and in the right are equivalent.

You are given the locations of harbours on the western and southern shores of Byteotia. Your task is to determine the size of the largest set of pairwise non-equivalent road networks.

Input

The first line of the input contains two integers n, m $(1 \le n, m \le 500)$, denoting the numbers of harbours on the western side and on the southern side, respectively. The second line contains n distinct integers $y_1, \ldots, y_n$ $(1 \le y_i \le 10^9)$ describing locations of the harbours on the western side: the i-th of these harbours is located at point $(0, y_i)$. The third line contains m distinct integers $x_1, \ldots, x_m$ $(1 \le x_j \le 10^9)$ describing locations of the harbours on the southern side: the j-th of these harbours is located at point $(x_j, 0)$.

Output

You should output the size of the largest set of pairwise nonequivalent roads networks, modulo $10^9 + 7$.

Example

standard input	standard output
2 2 1 2 1 2	13
8 9 39 58 64 23 72 66 80 30 93 23 33 72 79 48 19 92 98	914854829

Problem G. Reorganization

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Bytiana is a world-class human resources specialist. Recently, she was hired by an IT company to reorganize its decaying structure.

The brand-new structure proposed by Bytiana is ingenious in its simplicity. It can be easily described using a few simple terms: *manager*, *boss* and *superior*:

- Exactly one of the company workers becomes a *manager* and has no *boss*.
- Each of the other workers (that is, everyone excluding the *manager*) has exactly one *boss*.
- The set of a worker's *superiors* are his/her *boss* and his *boss' superiors*.
- The *manager* is a *superior* of all the other workers.

In the graph-theoretic language: a worker is a node in a hierarchy tree, the *manager* is its root, a worker's *boss* is a parent of the node corresponding to this worker and a *superior* is its ancestor.

Bytiana realized that the current problems in the company were originated in antipathies between the workers, one of which is the superior of the other. Bytiana wants to avoid such situations. She gathered information about the workers' preferences – the employees had submitted some opinions, each in one of two formats: "*I would like X to be my superior*" or "*I would like X not to be my superior*". Now Bytiana wants to create such a company structure, that all these preferences are fulfilled.

Input

The first line of the input contains two integers n, m ($1 \le n \le 1000$, $0 \le m \le \min\{n(n-1), 10\,000\}$) – the number of the company's workers and the number of collected preferences, respectively. The workers are numbered 1 through n.

The following m lines describe the preferences. Each of them is formatted a b c ($1 \le a, b \le n$, $a \ne b$, $c \in \{\texttt{T}, \texttt{N}\}$). If $c = \texttt{T}$, then a wants b to be his/her superior. If $c = \texttt{N}$, then a wants b **not** to be his/her superior. For each ordered pair (a, b), the line a b c occurs in the input at most once.

Output

If it is impossible to create the structure fulfilling all the preferences, output `NO`.

However, if it is possible, you should output n lines describing any structure fulfilling the preferences. The i-th of these lines, $i \ge 1$, should contain a single integer – the index of the i-th worker's boss. If the i-th worker is assigned to be the manager, the corresponding row should contain the number 0.

Example

standard input	standard output
4 4 4 1 T 4 2 T 3 2 N 4 3 N	0 1 1 2
2 2 1 2 N 2 1 N	NO

Problem H. Matching

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

A *matching* in a graph G is a subset M of its edges such that no two edges of M share an endpoint.

An unordered pair (u, v) of vertices of G ($u \neq v$) is called *promising* if and only if adding an edge (u, v) to G increases the size of the largest matching in G.

A connected graph G consisting of n vertices and $n-1$ edges is given. You need to determine the number of promising pairs of vertices of G.

Input

The first line of the input contains an integer n ($1 \leq n \leq 200\,000$) denoting the number of vertices of G. The vertices are numbered 1 through n.

The following $n-1$ lines contain descriptions of edges of G. The i-th of them contains two integers a_i, b_i ($1 \leq a_i, b_i \leq n$), denoting that the i-th edge connects vertices a_i and b_i. You can assume that G is connected.

Output

You should output a single number – the number of promising pairs of vertices in G.

Example

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

Note

The only promising pairs of vertices in the sample are $(3, 4)$, $(3, 5)$ and $(4, 5)$.

Problem I. Word

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

A word s is lexicographically earlier than a word t ($s \neq t$) if any of the following conditions hold:

- s is a prefix of t,
- there exists a number $m \geq 0$ such that the m-character prefixes of s and t are equal, and the $(m+1)$-st character of s is earlier in the alphabetical order than the $(m+1)$-st character of t.

Your task is to find the k-th non-empty word in the lexicographical order which consists of at most n characters from the set $\{\texttt{a}, \texttt{b}, \texttt{c}\}$ and whose every two adjacent characters are distinct.

Input

The first and only line of the input contains two integers n, k ($1 \leq n \leq 10^6$, $1 \leq k \leq 10^{18}$) described in the task statement.

Output

If there are less than k words fulfilling the conditions described in the task statement, your program should output `NO`. Otherwise, output the k-th word in the lexicographical order.

Example

standard input	standard output
3 7	acb
2 10	NO

Problem J. Tournament

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

The *Epic Warriors* final tournament is currenly held in Byteotia. n of best the competitors chosen in a remote elimination round are competing. The tournament will be conducted the following way: as long as there are at least two competitors, two of them will be chosen randomly. Next, the two will fight against each other and the loser will be permanently eliminated from the tournament. There are no ties in Epic Warrios, so there will be exactly $n - 1$ battles in total.

For some pairs of competitors (a, b), it is clear who will be the winner if a and b face each other in a fight. You need to determine which competitors have a possibility of winning tournament. A competitor a has a possibility of winning the tournament if there is a sequence of possible fights and their results such that a will win the tournament.

Input

The first line of the input contains an integer n ($2 \le n \le 500$), denoting the number of competitors in the tournament. The competitors are numbered 1 through n.

The i-th of the following n lines contains a sequence of n characters $c_{i,1}c_{i,2}\ldots c_{i,n}$. For every i, $c_{i,i} = \texttt{X}$ holds. Moreover, for $i \neq j$, we know that $c_{i,j} \in \{\texttt{W}, \texttt{P}, \texttt{?}\}$. $c_{i,j} = \texttt{W}$ means that competitor i will win the fight against competitor j; $c_{i,j} = \texttt{P}$ means that competitor i will lose the fight against competitor j; and $c_{i,j} = \texttt{?}$ means that both competitors i and j can possibly win the fight against each other.

Furthermore, $c_{i,j} = \texttt{W}$ holds if and only if $c_{j,i} = \texttt{P}$, and $c_{i,j} = \texttt{?}$ holds if and only if $c_{j,i} = \texttt{?}$.

Output

You should output the identifiers of the competitors that have a possibility of winning the tournament. The identifiers should be output in ascending order, one per line.

Example

standard input	standard output
4 XPPP WX?W W?XW WPPX	2 3

Problem K. Scale

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

Byteasar got a weighing scale for his thirteen birthday. He was very happy with this gift and immediately started weighing items that were around. However, it quickly turned out that the scale is not very precise and always displays mass rounded **down** to the nearest integer multiple of c grams. The scale has one more defect: if the mass of items situated on the weight is at least $k \cdot c$ grams, it reports an error and does not display any mass.

At first, Byteasar got sad that he will not be able to measure the exact masses of all his items. Still, he wants to learn as much as possible about the items using the fact that many items can be simultaneously put on the scale. This way Byteasar can get more information about his items: for example, this can be sufficient to determine for some pairs of items that one item is certainly heavier than the other.

Your task is to determine for how many pairs of items (x, y) Byteasar is able to determine using his scale that x is heavier than y. Luckily, the masses of Byteasar's items are integer multiplies of one gram. However, he does not know that – he only assumes that their masses are some real numbers of grams. Fortunately, he knows the values of k and c: he found them in the scale's manual.

Input

The first line of the input contains three integers n, k and c ($1 \le n, k \le 1\,000$, $1 \le c \le 5\,000$), denoting the number of Byteasar's items and the scale's parameters, respectively.

The scale measures weight with precision of c grams and displays mass rounded down to nearest integer multiple of c grams if and only if the items situated on it weigh strictly less than $k \cdot c$ grams. Otherwise it displays an error. The items are numbered 1 through n.

The second line of the input contains n integers $a_1, \ldots, a_n$ ($1 \le a_i < k \cdot c$). a_i denotes the mass of the item labelled i (in grams).

Output

You need to output the total number of pairs (x, y) such that Byteasar can deduce using his scale that x is certainly heavier than y.

Example

standard input	standard output
4 4 6 8 9 10 11	4

Note

Explanation of the sample:

Byteasar can deduce that the items in pairs $(1, 3)$, $(1, 4)$, $(2, 3)$ and $(2, 4)$ have different masses.