Problem C. Two paths

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

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>7 9
1 2
1 3
3 4
4 5
2 4
2 5
5 6
5 7
5 7</pre></td><td><pre>4
1 4 5 7</pre></td></tr>
</table>

Problem E. Epic battle

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

Bobby and Billy 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 Bobby, 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 Bobby's win, Billy's win or a draw. The winner of the final phase becomes the Grandmaster.

Your task is to check if Bobby has a winning strategy. We say that Bobby has a winning strategy if he can win (and become the Grandmaster) independently of Billy's moves in the first phase of the game. If Bobby cannot win when Billy plays optimally, you should check whether he is at least able to draw, that is, prevent Billy 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 Bobby's deck loses with the b-th Billy's deck. If $w = >$, then the a-th Bobby's deck wins with the b-th Billy'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 Bobby's deck draws with the b-th Billy'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 Bobby can win;
- `DRAW`, if Bobby cannot guarantee a win, but he can avoid losing;
- `LOSS` otherwise.

Example

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

Problem F. Reachability

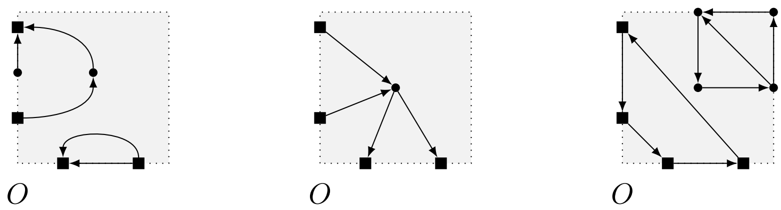
Input file: `standard input`
Output file: `standard output`
Time limit: 4 seconds
Memory limit: 512 mebibytes

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 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 $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ 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:	1 second
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, $c_{i,i} = \mathtt{X}$ holds. Moreover, for $i \neq j$, we know that $c_{i,j} \in \{\mathtt{W}, \mathtt{P}, \mathtt{?}\}$. $c_{i,j} = \mathtt{W}$ means that competitor i will win the fight against competitor j; $c_{i,j} = \mathtt{P}$ means that competitor i will lose the fight against competitor j; and $c_{i,j} = \mathtt{?}$ means that both competitors i and j can possibly win the fight against each other.

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

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 L. Blackjack

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

Blackjack is a game played with one or more decks of playing cards. The objective of each player is to have the score of the hand close to 21 without going over 21. The score is the total points of the cards in the hand. Points are assigned to cards in next way:

- Cards from 2 to 10 cost their value, i.e. 2 costs 2 points, 3 costs 3 points etc.
- Face cards (jack, queen and king) cost 10 points.
- An ace may cost 11 **or** 1 points in such a way the score gets closer to but does not exceed 21.

Some special types of hands are defined:

- A hand of more than 21 points is called **bust**. It causes a player to lose automatically.
- A hand of 21 points with exactly two cards, that is, a pair of an ace and a ten-point card (a face card or a ten) is called a **blackjack** and causes a player to win automatically.

Consider blackjack for one player (the dealer). She first deals two cards to himself. In case the dealer gets a blackjack, the dealer wins and the game ends with blackjack. Then dealer decides to take another card (hit) or to stop taking (stand) in his turn, and she plays according to the following rules:

- Hits if the summary score of the cards on her hand is 16 or less.
- Also hits if the summary score of the cards on her hand is 17 and one of aces is counted as 11.
- Stands otherwise.

Write a program that counts the score of the dealer's hand after his play for each given sequence of cards.

Input

The first line of the input contains an integer T ($1 \leq T \leq 100$) — number of test cases. Then T test cases follow.

Each test case consists of two lines. The first line contains two characters, which indicate the cards in the dealer's initial hand. The second line contains eight characters, which indicate the top eight cards in the pile after all players finish their plays.

A character that represents a card is one of `A`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `T`, `J`, `Q` and `K`, where `A` is an ace, `T` a ten, `J` a jack, `Q` a queen and `K` a king.

Characters in a line are delimited by a single space.

The same cards may appear up to four times in one test case. Note that it is impossible that dealer will use more than eight cards from the stack.

Output

For each test case, print "`blackjack`" if the dealer has a blackjack; "`bust`" if the dealer busts. Otherwise print the summary score of the dealer's hand.

Example

standard input	standard output
4 4 6 7 4 A J T K 5 6 A J 2 3 9 Q Q 2 3 9 T 3 8 2 3 2 J J A A 2 2 2 4 3 J 2 3 4 K	17 blackjack 21 bust

Problem M. Conquerors

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

On the planet Pluk at the year 0 several countries exist. They are conquering each other, until only one country will remain. You are chosen to be king of Nowhereland, and your country is not the biggest one. So your goal is to keep your country as long as it possible.

Each year, each country that is still alive when it is its turn will conquer one other country, and more specifically, all countries except Nowhereland will always conquer the largest country smaller than self. If there is no smaller country, then a country will not conquer anyone. Countries do this in order of decreasing size. It is guaranteed that at the start all countries had different sizes/

As king of Nowhereland, you decided to break this law and may conquer **any** country smaller than yours (or to not conquer anyone, if you prefer to). You still can never conquer the country larger than yours. Note that if after your decision two countries will have the same size, then you will be accused in breaking of the laws and forced to resign.

When an country conquers another country, its new size will be the sum of their previous sizes. As an illustration, suppose that Nowhereland is the 5th largest country on Pluk initially. First, the largest one will conquer the second largest one, resulting in a country of the sum of their sizes. As a result of being conquered, the 2nd largest country does not get to conquer anyone, and the 3rd largest country will conquer the 4th largest, producing a country with the sum of their sizes. Now it's your turn. After everyone has had their turn, we start again at the beginning, with the order determined by the new sizes.

Your goal is to let Nowhereland survive as many rounds as you can without being forced to resign.

Input

The first line of the input contains two integers n and m with $2 \leq n \leq 50$ and $2 \leq m \leq n$. n is the number of initial countries, and m is the number of the Nowhereland. This is followed by a line with n integers $10^6 > s_1 > s_2 > s_3 > \ldots > s_n \geq 0$; s_i is the size of the $i-th$ country.

Output

Print the year when Nowhereland will be conquerred, if you will follow an optimal strategy.

Example

standard input	standard output
10 3 100 95 66 53 51 50 49 19 15 12	2
10 4 100 95 66 53 51 50 49 19 15 12	1

Problem N. Inverse LCS

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

Consider next algorithm to find *least common subsequence* of integer sequences a and b.

Lets $a = \{a_1, a_2, \ldots, a_n\}$, $b = \{b_1, b_2, \ldots, b_m\}$.

$$\begin{aligned} f(i,j) &= \begin{cases} max(f(i-1,j), f(i,j-1), f(i-1,j-1)+1), \text{if } a_i = b_j \\ max(f(i-1,j), f(i,j-1)), \text{else.} \end{cases} \\ f(i,0) &= 0 \\ f(0,j) &= 0 \end{aligned}$$

You are given the matrix f, produced by this algoritm. Your goal is to find possible initial sequences a and b for this matrix.

Note that subsequence $x = \langle x_1, x_2, \ldots, x_n \rangle$ is called *subsequence* $y = \langle y_1, y_2, \ldots, y_m \rangle$, if exists increasing sequence of indices $1 \le i_1 < i_2 < \ldots i_n \le m$ such as for any $j = 1, 2, \ldots, n$ $y_{i_j} = x_j$.

Input

First line of the input contains two integers n and m — lengths of the sequences a and b, respectively ($1 \le n, m \le 500$). Next n lines contain description of the matrix f, $f(i,j), i \in [1..n], j \in [1..m]$.

It is guaranteed that input data are correct and at least one pair of sequences a and b give matrix f as the result of work of the given algorithm.

Output

First line of the output must contain n positive integers — sequence a, second line must contain m positive integers — sequence b. If more than one answers are possible, print any of them.

Example

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

Problem O. Quardilateral

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

You have given two points A and B such as $0 < x_a, y_a, x_b, y_b \leq 10^4$ and $(x_a - x_b) \cdot (y_a - y_b) < 0$. Build two points C and D such as:

- quadrilateral $ABCD$ is convex;
- exactly one of points C and D lies on the x-axis, and exactly one lies on the y-axis;
- perimeter of the quadrilateral must be minimum possible.
- all points A, B, C and D are pairwise distinct.

Then you should print the resulting perimeter.

Input

Input contains four integers x_a, y_a, x_b and y_b — coordinates of points A and B respectively ($0 < x_a, y_a, x_b, y_b \leq 5000$, $(x_a - x_b) \cdot (y_a - y_b) < 0$.

Output

Print the perimeter with absolute error 10^{-3} or better.

Example

standard input	standard output
3 2 1 4	10.04

Problem P. Fill the Square

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

You need fill the square $n \times n$ with integers from 1 to n^2 in next way: fill first line of square with integers from 1 to n, then fill second line of square with integers from $n+1$ to $2n$, listed **from left to right**, then fill third line of square with integers from $2n+1$ to $3n$ listed in normal order etc (i.e. if lines are 1-indexed, numbers in odd-indexed lines are listed from left to right, numbers in even-indexed lines are listed from right to left).

Input

Input constists of one integer n ($1 \le n \le 100$).

Output

Print n lines, each containing n integers — answer to the problem.

Example

standard input	standard output
5	1 2 3 4 5 10 9 8 7 6 11 12 13 14 15 20 19 18 17 16 21 22 23 24 25

Problem Q. Triangles

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

Consider integer K. We call integer A K-*interesting*, if for all integer $B \geq A$ exists exactly K distinct ingegers C such as triangle with sides A, B and C exists.

Given K, find number of K-interesting integers.

Input

Input consists of one integer K ($0 \leq K \leq 10^{18}$).

Output

Print the answer to the problem — number of K-interesting integers for a given K.

Example

standard input	standard output
1	1
2	0