

Problem A. Topological Sorting

Time limit	500 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

Sandro is a well organised person. Every day he makes a list of things which need to be done and enumerates them from 1 to n . However, some things need to be done before others. In this task you have to find out whether Sandro can solve all his duties and if so, print the correct order.

Input

In the first line you are given an integer n and m ($1 \leq n \leq 10000$, $1 \leq m \leq 1000000$). On the next m lines there are two distinct integers x and y , ($1 \leq x, y \leq 10000$) describing that job x needs to be done before job y .

Output

Print "Sandro fails." if Sandro cannot complete all his duties on the list. If there is a solution print the correct ordering, the jobs to be done separated by a space. If there are multiple solutions print the one, whose first number is smallest, if there are still multiple solutions, print the one whose second number is smallest, and so on.

Example 1

Input	Output
8 9 1 4 1 2 4 2 4 3 3 2 5 2 3 5 8 2 8 6	1 4 3 5 7 8 2 6

Example 2

Input	Output
2 2 1 2 2 1	Sandro fails.

Problem B. Topological Sorting

Time limit	10000 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

The problem is the same as [TOPOSORT](#), but the limits are different and the tests are different. The differences are highlighted below. Now, the TOPOSORT problem has only solutions in C/++ so it seems impossible to solve it in other languages, for silly reasons. With the tests here, you should still need to find sub-quadratic solutions, but you may do so in languages with I/O that isn't super-fast.

So, the problem: You are given a directed graph, and you should find the [lexicographically minimal topological sort](#) of it.

The first line of input has the number n of vertices and the number m of arcs. We have $0 \leq m, n \leq 200000$. The vertices are the numbers $1, \dots, n$. The next m numbers each list one arc by giving the source and the target vertex. If there is no cycle, you should output a permutation of the vertices, on one line, whitespace-separated. If there is a cycle, you should print one line containing "Sandro fails." (dot included).

Problem C. Dependency Problems

Time limit	1000 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

We bought a brand new computer and now we would like to install an operating system. The only problem is that our chosen operating system consists of many packages and they cannot be installed in an arbitrary order. E.g. you cannot install the package tuxracer, which depends on the package libSDL, before you install libSDL. But libSDL can depend on another packages and so on. The packages may only be installed one at a time. You may install a package only if you already installed all packages it depends on. Your task is to determine how many packages can be installed on our computer.

Input file specification

The input file contains a single line for each available package. The line for each package P begins with the name of the package. The name of each package is a non-empty string of printable characters containing no spaces. Following the name of the package P is the dependency list of P. The dependency list is simply a list of names of packages that P depends on, separated by spaces. A whitespace followed by a single 0 (zero) is at the end of each line. You may assume that no package has the name 'o'.

The dependency list of a package P may be empty; in that case, P does not depend on any packages and may be installed immediately. It is possible that a package Q occurs in the dependency list of a package P more than once; this merely means that P depends on Q, nothing more. Only the packages that have a dependency list in the input file are available and may be installed. It is possible that a package P depends on a package that is not available. Such a package cannot be installed.

The number of lines in the input file will be less than 9000.

Output file specification

The output consists of one number -- the maximum number of packages that may be installed on the computer.

Example

Input	Output
a b c b 0 b c 0 c 0 d e f 0 e f 0 f e 0 g h 0	3

Note

Package c can be installed immediately. Package b depends only on c, and hence can also be installed once we installed c. Finally, package a depends only on b and c and can now also be installed. It is easy to verify that no other packages can be installed.

Problem D. Project File Dependencies

Time limit	405 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

Project managers, such as the UNIX utility `make`, are used to maintain large software projects made up from many components. Users write a *project file* specifying which components (called *tasks*) depend on others and the project manager can automatically update the components in the correct order.

Problem

Write a program that reads a project file and outputs the order in which the tasks should be performed.

Input

For simplicity we represent each task by an integer number from $1, 2, \dots, N$ (where N is the total number of tasks). The first line of input specifies the number N of tasks and the number M of rules, such that $N \leq 100$, $M \leq 100$.

The rest of the input consists of M rules, one in each line, specifying dependencies using the following syntax:

$$T_0 \ k \ T_1 \ T_2 \ \dots \ T_k$$

This rule means that task number T_0 depends on k tasks T_1, T_2, \dots, T_k (we say that task T_0 is the *target* and T_1, \dots, T_k are *dependents*).

Note that tasks numbers are separated by single spaces and that rules end with a newline. Rules can appear in any order, but each task can appear as target only once.

Your program can assume that there are no circular dependencies in the rules, i.e. no task depends directly or indirectly on itself.

Output

The output should be a single line with the permutation of the tasks $1 \dots N$ to be performed, ordered by dependencies (i.e. no task should appear before others that it depends on).

To avoid ambiguity in the output, tasks that do not depend on each other should be ordered by their number (lower numbers first).

Example

Input	Output
5 4 3 2 1 5 2 2 5 3 4 1 3 5 1 1	1 5 3 2 4

Problem E. Ordering Tasks

Time limit 3000 ms

OS Linux

Statement

John has n tasks to do. Unfortunately, the tasks are not independent and the execution of one task is only possible if other tasks have already been executed.

Input

The input will consist of several instances of the problem. Each instance begins with a line containing two integers, $1 \leq n \leq 100$ and m . n is the number of tasks (numbered from 1 to n) and m is the number of direct precedence relations between tasks. After this, there will be m lines with two integers i and j , representing the fact that task i must be executed before task j .

An instance with $n = m = 0$ will finish the input.

Output

For each instance, print a line with n integers representing the tasks in a possible order of execution.

Input

5 4

1 2

2 3

1 3

1 5

0 0

Output

1 4 2 5 3

Searching and sorting are part of the theory and practice of computer science. For example, binary search provides a good example of an easy-to-understand algorithm with sub-linear complexity. Quicksort is an efficient $O(n \log n)$ [average case] comparison based sort.

KWIC-indexing is an indexing method that permits efficient “human search” of, for example, a list of titles.

Given a list of titles and a list of “words to ignore”, you are to write a program that generates a KWIC (Key Word In Context) index of the titles. In a KWIC-index, a title is listed once for each keyword that occurs in the title. The KWIC-index is alphabetized by keyword.

Any word that is not one of the “words to ignore” is a potential keyword.

For example, if words to ignore are “the, of, and, as, a” and the list of titles is:

```
Descent of Man
The Ascent of Man
The Old Man and The Sea
A Portrait of The Artist As a Young Man
```

A KWIC-index of these titles might be given by:

```
          a portrait of the ARTIST as a young man
                        the ASCENT of man
                          DESCENT of man
                    descent of MAN
                the ascent of MAN
                    the old MAN and the sea
a portrait of the artist as a young MAN
                        the OLD man and the sea
                          a PORTRAIT of the artist as a young man
                    the old man and the SEA
a portrait of the artist as a YOUNG man
```

Input

The input is a sequence of lines, the string ‘:.’ is used to separate the list of words to ignore from the list of titles. Each of the words to ignore appears in lower-case letters on a line by itself and is no more than 10 characters in length. Each title appears on a line by itself and may consist of mixed-case (upper and lower) letters. Words in a title are separated by whitespace. No title contains more than 15 words.

There will be no more than 50 words to ignore, no more than than 200 titles, and no more than 10,000 characters in the titles and words to ignore combined. No characters other than ‘a’–‘z’, ‘A’–‘Z’, and white space will appear in the input.

Output

The output should be a KWIC-index of the titles, with each title appearing once for each keyword in the title, and with the KWIC-index alphabetized by keyword. If a word appears more than once in a title, each instance is a potential keyword.

The keyword should appear in all upper-case letters. All other words in a title should be in lower-case letters. Titles in the KWIC-index with the same keyword should appear in the same order as they appeared in the input file. In the case where multiple instances of a word are keywords in the same title, the keywords should be capitalized in left-to-right order.

Case (upper or lower) is irrelevant when determining if a word is to be ignored.

The titles in the KWIC-index need NOT be justified or aligned by keyword, all titles may be listed left-justified.

Sample Input

```
is
the
of
and
as
a
but
:
Descent of Man
The Ascent of Man
The Old Man and The Sea
A Portrait of The Artist As a Young Man
A Man is a Man but Bubblesort IS A DOG
```

Sample Output

```
a portrait of the ARTIST as a young man
the ASCENT of man
a man is a man but BUBBLESORT is a dog
DESCENT of man
a man is a man but bubblesort is a DOG
descent of MAN
the ascent of MAN
the old MAN and the sea
a portrait of the artist as a young MAN
a MAN is a man but bubblesort is a dog
a man is a MAN but bubblesort is a dog
the OLD man and the sea
a PORTRAIT of the artist as a young man
the old man and the SEA
a portrait of the artist as a YOUNG man
```

A rare book collector recently discovered a book written in an unfamiliar language that used the same characters as the English language. The book contained a short index, but the ordering of the items in the index was different from what one would expect if the characters were ordered the same way as in the English alphabet. The collector tried to use the index to determine the ordering of characters (i.e., the collating sequence) of the strange alphabet, then gave up with frustration at the tedium of the task.

You are to write a program to complete the collector's work. In particular, your program will take a set of strings that has been sorted according to a particular collating sequence and determine what that sequence is.

Input

The input consists of an ordered list of strings of uppercase letters, one string per line. Each string contains at most 20 characters. The end of the list is signalled by a line with the single character '#'. Not all letters are necessarily used, but the list will imply a complete ordering among those letters that are used.

Output

Your output should be a single line containing uppercase letters in the order that specifies the collating sequence used to produce the input data file.

Sample Input

```
XWY
ZX
ZXY
ZXW
YWWX
#
```

Sample Output

```
XZYW
```

Problem H. Fox And Names

Time limit 2000 ms

Mem limit 262144 kB

Input file `stdin`

Output file `stdout`

Fox Ciel is going to publish a paper on FOCS (Foxes Operated Computer Systems, pronounce: "Fox"). She heard a rumor: the authors list on the paper is always sorted in the lexicographical order.

After checking some examples, she found out that sometimes it wasn't true. On some papers authors' names weren't sorted in lexicographical order in normal sense. But it was always true that after some modification of the order of letters in alphabet, the order of authors becomes lexicographical!

She wants to know, if there exists an order of letters in Latin alphabet such that the names on the paper she is submitting are following in the lexicographical order. If so, you should find out any such order.

Lexicographical order is defined in following way. When we compare s and t , first we find the leftmost position with differing characters: $s_i \neq t_i$. If there is no such position (i. e. s is a prefix of t or vice versa) the shortest string is less. Otherwise, we compare characters s_i and t_i according to their order in alphabet.

Input

The first line contains an integer n ($1 \leq n \leq 100$): number of names.

Each of the following n lines contain one string $name_i$ ($1 \leq |name_i| \leq 100$), the i -th name.

Each name contains only lowercase Latin letters. All names are different.

Output

If there exists such order of letters that the given names are sorted lexicographically, output any such order as a permutation of characters 'a'–'z' (i. e. first output the first letter of the modified alphabet, then the second, and so on).

Otherwise output a single word "Impossible" (without quotes).

Examples

Input	Output
3 rivest shamir adleman	bcdefghijklmnopqrstuvwxyz

Input	Output
10 tourist petr wjmzbr yeputons vepifanov scottwu oooooooooooooooooooo subscriber rowdark tankengineer	Impossible

Input	Output
10 petr egor endagorion feferivan ilovetanyaromanova kostka dmitriyh maratsnowbear bredorjaguarturnik cgyforever	aghjlnopefikdmbcqrstuvwxyz

Input	Output
7 car care careful carefully becarefuldontforgetsomething otherwiseyouwillbehacked goodluck	acdefghijklmnopqrstuvwxyz

Problem I. Answer the boss!

Time limit	1000 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

Eloy is a hard worker man, however, he is constantly bullied by his superiors, molested by this, one day he was wondering in what “rank” you are, so you can bully the people with lower ranks, also to discover who can really bully Eloy!.

Now, given the number of employees and the number of relations between them, Eloy need you to output the “rank” which employee is in, being 1 the “boss” (not bullied by anybody) and the employee who are in these ranks

Input

There will be an integer T denoting the number of test cases, then, T test cases will follow. Each test case starts with two integers N and R, the number of employees and the number of relations between them. The next R lines consists of two integers R1 and R2, meaning that “employee R1 is lower than employee R2's rank”.

Output

You will output for each test case the string “Scenario #i:” where i is the test case you are analyzing, after that, you will print N lines, for each line you will output the rank of the employee and the employee itself, if there is the same rank for several employees, then output them lexicographically ordered (the first is the lower.)

Sample

Input	Output
2 5 6 2 0 2 4 1 4 1 2 3 2 4 0 5 4 1 0 2 0 3 2 4 2	Scenario #1: 1 0 2 4 3 2 4 1 4 3 Scenario #2: 1 0 2 1 2 2 3 3 3 4

Blank line between test cases for clarification and separation. Please note that can be more than one "boss" (not bullied by anybody.)

Constraints

$1 \leq N \leq 1000$; $1 \leq R \leq 10000$

Problem J. Course Schedule

Time limit 1000 ms

Mem limit 524288 kB

You have to complete n courses. There are m requirements of the form "course a has to be completed before course b ". Your task is to find an order in which you can complete the courses.

Input

The first input line has two integers n and m : the number of courses and requirements. The courses are numbered $1, 2, \dots, n$.

After this, there are m lines describing the requirements. Each line has two integers a and b : course a has to be completed before course b .

Output

Print an order in which you can complete the courses. You can print any valid order that includes all the courses.

If there are no solutions, print "IMPOSSIBLE".

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 2 \cdot 10^5$
- $1 \leq a, b \leq n$

Example

Input	Output
5 3 1 2 3 1 4 5	3 4 1 5 2

Problem K. Longest Flight Route

Time limit 1000 ms

Mem limit 524288 kB

Uolevi has won a contest, and the prize is a free flight trip that can consist of one or more flights through cities. Of course, Uolevi wants to choose a trip that has as many cities as possible.

Uolevi wants to fly from Syrjälä to Lehmälä so that he visits the maximum number of cities. You are given the list of possible flights, and you know that there are no directed cycles in the flight network.

Input

The first input line has two integers n and m : the number of cities and flights. The cities are numbered $1, 2, \dots, n$. City 1 is Syrjälä, and city n is Lehmälä.

After this, there are m lines describing the flights. Each line has two integers a and b : there is a flight from city a to city b . Each flight is a one-way flight.

Output

First print the maximum number of cities on the route. After this, print the cities in the order they will be visited. You can print any valid solution.

If there are no solutions, print "IMPOSSIBLE".

Constraints

- $2 \leq n \leq 10^5$
- $1 \leq m \leq 2 \cdot 10^5$
- $1 \leq a, b \leq n$

Example

Input	Output
5 5 1 2 2 5 1 3 3 4 4 5	4 1 3 4 5

Problem L. Game Routes

Time limit 1000 ms
Mem limit 524288 kB

A game has n levels, connected by m teleporters, and your task is to get from level 1 to level n . The game has been designed so that there are no directed cycles in the underlying graph. In how many ways can you complete the game?

Input

The first input line has two integers n and m : the number of levels and teleporters. The levels are numbered $1, 2, \dots, n$.

After this, there are m lines describing the teleporters. Each line has two integers a and b : there is a teleporter from level a to level b .

Output

Print one integer: the number of ways you can complete the game. Since the result may be large, print it modulo $10^9 + 7$.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 2 \cdot 10^5$
- $1 \leq a, b \leq n$

Example

Input	Output
<pre> 4 5 1 2 2 4 1 3 3 4 1 4 </pre>	<pre> 3 </pre>

Problem M. Quantum Superposition

Time limit 1000 ms

Mem limit 1048576 kB

OS Linux

Through quantum superposition, you are in two universes at the same time. Each universe can be represented as a directed acyclic graph. You begin at the start vertices in each universe and your goal is to reach the end vertices. Now, you are wondering, can you reach the end vertices in both universes where the sum of steps made in each universe is exactly equal to a given number?

Input

The first line contains $1 \leq N_1, N_2 \leq 1\,000$, the number of vertices in each universe, and $0 \leq M_1, M_2 \leq 2\,000$, the number of edges in each universe. In both universes, vertex 1 is the start vertex. Vertices N_1 and N_2 are the end vertices in each respective universe.

Each of the next M_1 lines contains two numbers $1 \leq u, v \leq N_1$, indicating a directed edge from vertex u to vertex v in universe 1.

Each of the next M_2 lines contains two numbers $1 \leq u, v \leq N_2$, indicating a directed edge from vertex u to vertex v in universe 2.

There will not be duplicate edges. It is guaranteed that the universes are acyclic, and there exists a path from start to end vertex in each universe.

The next line contains $1 \leq Q \leq 2\,000$, the number of queries. Each query consists of a line with a single integer $0 \leq q \leq 2\,000$ that is the sum of steps made in each universe.

Output

For each query, output “Yes” if it is possible to reach the end vertices in both universes where the sum of steps made in each universe is exactly equal to the given query, or “No” otherwise.

Sample 1

Input	Output
3 2 3 1	No
1 2	No
1 3	Yes
2 3	Yes
1 2	No
5	
0	
1	
2	
3	
4	

Problem N. Timeline

Input file `timeline.in`

Output file `timeline.out`

Bessie attended N milking sessions ($1 \leq N \leq 10^5$) over the past M days ($2 \leq M \leq 10^9$). However, she is having trouble remembering when she attended each session.

For each session $i = 1 \dots N$, she knows that it occurred no earlier than day S_i ($1 \leq S_i \leq M$). Additionally, Bessie has C memories ($1 \leq C \leq 10^5$), each described by a triple (a, b, x) , where she recalls that session b happened at least x days after a .

Help Bessie by computing the earliest possible date of occurrence for each milking session. It is guaranteed that Bessie did not remember incorrectly; in other words, there exists an assignment of sessions to days in the range $1 \dots M$ such that all constraints from her memories are satisfied.

SCORING:

- Test cases 2-4 satisfy $N, C \leq 10^3$.
- Test cases 5-10 satisfy no additional constraints.

INPUT FORMAT (file `timeline.in`):

The first line of input contains N , M , and C .

The next line contains N space-separated integers S_1, S_2, \dots, S_N . Each is in the range $1 \dots M$.

The next C lines contain three integers, a , b , and x indicating that session b happened at least x days after a . For each line, $a \neq b$, a and b are in the range $1 \dots N$, and x is in the range $1 \dots M$.

OUTPUT FORMAT (file `timeline.out`):

Output N lines giving the earliest possible date of occurrence for each session.

Input	Output
4 10 3	1
1 2 3 4	6
1 2 5	3
2 4 2	8
3 4 4	

Session two occurred at least five days after session one, so it cannot have occurred before day $1 + 5 = 6$. Session four occurred at least two days after session two, so it cannot have occurred before day $6 + 2 = 8$.

Problem credits: Mark Gordon

Problem O. Substring

Time limit 3000 ms

Mem limit 262144 kB

You are given a graph with n nodes and m **directed** edges. One lowercase letter is assigned to each node. We define a path's value as the number of the most frequently occurring letter. For example, if letters on a path are "abaca", then the value of that path is 3. Your task is find a path whose value is the largest.

Input

The first line contains two positive integers n, m ($1 \leq n, m \leq 300\,000$), denoting that the graph has n nodes and m directed edges.

The second line contains a string s with only lowercase English letters. The i -th character is the letter assigned to the i -th node.

Then m lines follow. Each line contains two integers x, y ($1 \leq x, y \leq n$), describing a directed edge from x to y . Note that x can be equal to y and there can be multiple edges between x and y . Also the graph can be not connected.

Output

Output a single line with a single integer denoting the largest value. If the value can be arbitrarily large, output -1 instead.

Examples

Input	Output
5 4 abaca 1 2 1 3 3 4 4 5	3

Input	Output
6 6 xzyabc 1 2 3 1 2 3 5 4 4 3 6 4	-1

Input	Output
10 14 xzyzyzyzqx 1 2 2 4 3 5 4 5 2 6 6 8 6 5 2 10 3 9 10 9 4 6 1 10 2 8 3 7	4

Note

In the first sample, the path with largest value is $1 \rightarrow 3 \rightarrow 4 \rightarrow 5$. The value is 3 because the letter 'a' appears 3 times.

Problem P. Directing Edges

Time limit 3000 ms

Mem limit 262144 kB

You are given a graph consisting of n vertices and m edges. It is not guaranteed that the given graph is connected. Some edges are already directed and you can't change their direction. Other edges are undirected and you have to choose some direction for all these edges.

You have to direct undirected edges in such a way that the resulting graph is directed and acyclic (i.e. the graph with all edges directed and having no directed cycles). Note that you have to direct **all** undirected edges.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \leq t \leq 2 \cdot 10^4$) — the number of test cases. Then t test cases follow.

The first line of the test case contains two integers n and m ($2 \leq n \leq 2 \cdot 10^5$, $1 \leq m \leq \min(2 \cdot 10^5, \frac{n(n-1)}{2})$) — the number of vertices and the number of edges in the graph, respectively.

The next m lines describe edges of the graph. The i -th edge is described with three integers t_i , x_i and y_i ($t_i \in [0; 1]$, $1 \leq x_i, y_i \leq n$) — the type of the edge ($t_i = 0$ if the edge is undirected and $t_i = 1$ if the edge is directed) and vertices this edge connects (the undirected edge connects vertices x_i and y_i and directed edge is going from the vertex x_i to the vertex y_i). It is guaranteed that the graph do not contain self-loops (i.e. edges from the vertex to itself) and multiple edges (i.e. for each pair (x_i, y_i) there are no other pairs (x_i, y_i) or (y_i, x_i))).

It is guaranteed that both sum n and sum m do not exceed $2 \cdot 10^5$ ($\sum n \leq 2 \cdot 10^5$; $\sum m \leq 2 \cdot 10^5$).

Output

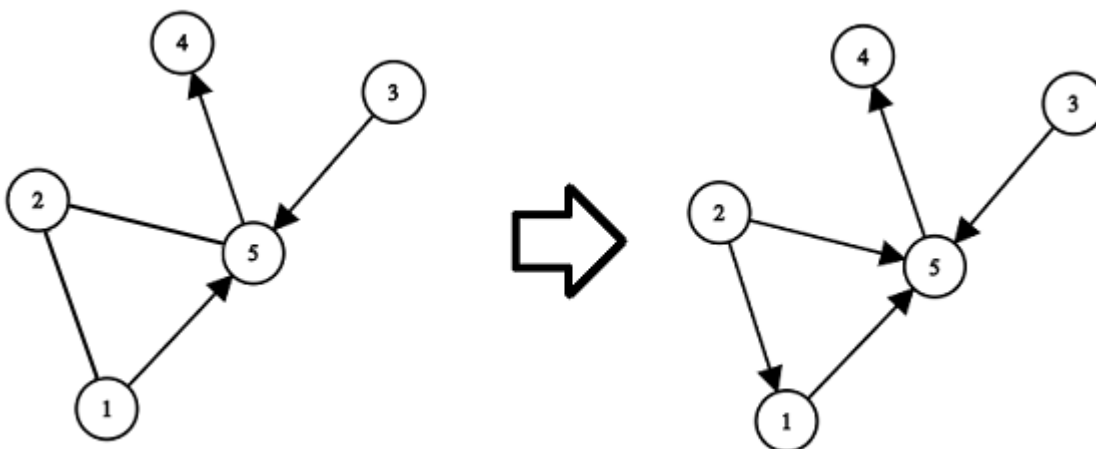
For each test case print the answer — "NO" if it is impossible to direct undirected edges in such a way that the resulting graph is directed and acyclic, otherwise print "YES" on the first line and m lines describing edges of the resulted directed acyclic graph (in any order). Note that you cannot change the direction of the already directed edges. If there are several answers, you can print any.

Examples

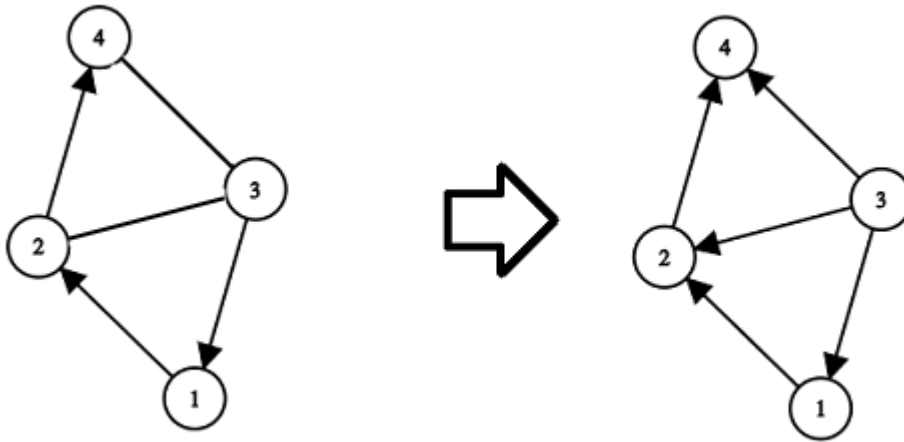
Input	Output
4 3 1 0 1 3 5 5 0 2 1 1 1 5 1 5 4 0 5 2 1 3 5 4 5 1 1 2 0 4 3 1 3 1 0 2 3 1 2 4 4 5 1 4 1 1 1 3 0 1 2 1 2 4 1 3 2	YES 3 1 YES 2 1 1 5 5 4 2 5 3 5 YES 1 2 3 4 3 1 3 2 2 4 NO

Note

Explanation of the second test case of the example:



Explanation of the third test case of the example:



Problem Q. Milking Order

Input file `milkorder.in`

Output file `milkorder.out`

Farmer John's N cows ($1 \leq N \leq 10^5$), numbered $1 \dots N$ as always, happen to have too much time on their hooves. As a result, they have worked out a complex social hierarchy related to the order in which Farmer John milks them every morning.

After weeks of study, Farmer John has made M observations about his cows' social structure ($1 \leq M \leq 50,000$). Each observation is an ordered list of some of his cows, indicating that these cows should be milked in the same order in which they appear in this list. For example, if one of Farmer John's observations is the list 2, 5, 1, Farmer John should milk cow 2 sometime before he milks cow 5, who should be milked sometime before he milks cow 1.

Farmer John's observations are prioritized, so his goal is to maximize the value of X for which his milking order meets the conditions outlined in the first X observations. If multiple milking orders satisfy these first X conditions, Farmer John believes that it is a longstanding tradition that cows with lower numbers outrank those with higher numbers, so he would like to milk the lowest-numbered cows first. More formally, if multiple milking orders satisfy these conditions, Farmer John would like to use the lexicographically smallest one. An ordering x is lexicographically smaller than an ordering y if for some j , $x_i = y_i$ for all $i < j$ and $x_j < y_j$ (in other words, the two orderings are identical up to a certain point, at which x is smaller than y).

Please help Farmer John determine the best order in which to milk his cows.

INPUT FORMAT (file `milkorder.in`):

The first line contains N and M . The next M lines each describe an observation. Line $i + 1$ describes observation i , and starts with the number of cows m_i listed in the observation followed by the list of m_i integers giving the ordering of cows in the observation. The sum of the m_i 's is at most 200,000.

OUTPUT FORMAT (file `milkorder.out`):

Output N space-separated integers, giving a permutation of $1 \dots N$ containing the order in which Farmer John should milk his cows.

Input	Output
4 3 3 1 2 3 2 4 2 3 3 4 1	1 4 2 3

Here, Farmer John has four cows and should milk cow 1 before cow 2 and cow 2 before cow 3 (the first observation), cow 4 before cow 2 (the second observation), and cow 3 before cow 4 and cow 4 before cow 1 (the third observation). The first two observations can be satisfied simultaneously, but Farmer John cannot meet all of these criteria at once, as to do so would require that cow 1 come before cow 3 and cow 3 before cow 1.

This means there are two possible orderings: 1 4 2 3 and 4 1 2 3, the first being lexicographically smaller.

Problem credits: Jay Leeds

Problem R. Pattern Matching

Time limit 2000 ms

Mem limit 262144 kB

You are given n patterns p_1, p_2, \dots, p_n and m strings s_1, s_2, \dots, s_m . Each pattern p_i consists of k characters that are either lowercase Latin letters or wildcard characters (denoted by underscores). All patterns are pairwise distinct. Each string s_j consists of k lowercase Latin letters.

A string a matches a pattern b if for each i from 1 to k either b_i is a wildcard character or $b_i = a_i$.

You are asked to rearrange the patterns in such a way that the first pattern the j -th string matches is $p[mt_j]$. You are allowed to leave the order of the patterns unchanged.

Can you perform such a rearrangement? If you can, then print any valid order.

Input

The first line contains three integers n, m and k ($1 \leq n, m \leq 10^5, 1 \leq k \leq 4$) — the number of patterns, the number of strings and the length of each pattern and string.

Each of the next n lines contains a pattern — k characters that are either lowercase Latin letters or underscores. All patterns are pairwise distinct.

Each of the next m lines contains a string — k lowercase Latin letters, and an integer mt ($1 \leq mt \leq n$) — the index of the first pattern the corresponding string should match.

Output

Print "NO" if there is no way to rearrange the patterns in such a way that the first pattern that the j -th string matches is $p[mt_j]$.

Otherwise, print "YES" in the first line. The second line should contain n distinct integers from 1 to n — the order of the patterns. If there are multiple answers, print any of them.

Examples

Input	Output
5 3 4 _b_d __b_ aaaa ab__ _bcd abcd 4 abba 2 dbcd 5	YES 3 2 4 5 1

Input	Output
1 1 3 __c cba 1	NO

Input	Output
2 2 2 a_ _b ab 1 ab 2	NO

Note

The order of patterns after the rearrangement in the first example is the following:

- aaaa
- __b_
- ab__
- _bcd
- _b_d

Thus, the first string matches patterns $ab_$, $_bcd$, $_b_d$ in that order, the first of them is $ab_$, that is indeed $p[4]$. The second string matches $_b_$ and $ab_$, the first of them is $_b_$, that is $p[2]$. The last string matches $_bcd$ and $_b_d$, the first of them is $_bcd$, that is $p[5]$.

The answer to that test is not unique, other valid orders also exist.

In the second example cba doesn't match $_c$, thus, no valid order exists.

In the third example the order (a_, _b) makes both strings match pattern 1 first and the order (_b, a_) makes both strings match pattern 2 first. Thus, there is no order that produces the result 1 and 2.

Problem S. Course Schedule II

Time limit 1000 ms

Mem limit 524288 kB

You want to complete n courses that have requirements of the form "course a has to be completed before course b ".

You want to complete course 1 as soon as possible. If there are several ways to do this, you want then to complete course 2 as soon as possible, and so on.

Your task is to determine the order in which you complete the courses.

Input

The first input line has two integers n and m : the number of courses and requirements. The courses are numbered $1, 2, \dots, n$.

Then, there are m lines describing the requirements. Each line has two integers a and b : course a has to be completed before course b .

You can assume that there is at least one valid schedule.

Output

Print one line having n integers: the order in which you complete the courses.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 2 \cdot 10^5$
- $1 \leq a, b \leq n$

Example

Input	Output
4 2 2 1 2 3	2 1 3 4

Problem T. Dumae

Time limit 3000 ms

Mem limit 1048576 kB

OS Windows

Do you know *Dumae*? It is a nickname of the most famous restaurant nearby KAIST, *Dumae Charcoal-grilled Barbecue*. Because *Dumae* is a very famous restaurant, lots of KAIST students stand in line even though it has not opened yet. Students wonder how long they have to wait, so they started to guess their order.

There are N students in waiting line and each of them has a distinct student ID from 1 to N . Student i (student with student ID i) guessed that he/she is either L_i -th, $(L_i + 1)$ -th, ..., $(R_i - 1)$ -th, or R_i -th person in the line. (i.e. the number of people standing relatively in front of him/her is in the interval $[L_i - 1, R_i - 1]$) Also, M claims are made, of which the i -th says that student v_i can see student u_i in the waiting line. It means student u_i is relatively in front of student v_i .

You wonder if all of students' guesses and claims were right. Find an order of waiting line that satisfies all the guesses and claims, or report that such an order does not exist.

Input

The first line contains two space-separated integers N, M .

$(1 \leq N \leq 300\,000, 0 \leq M \leq 1\,000\,000)$

In the next N lines, two space-separated integers L_i, R_i are given. $(1 \leq L_i \leq R_i \leq N)$

In the next M lines, two space-separated integers u_i, v_i are given. $(1 \leq u_i \leq N, 1 \leq v_i \leq N, u_i \neq v_i)$

Output

If there is no answer that satisfies the condition, print -1.

Otherwise, print N lines. In the i -th line, print the student ID of the i -th student from the front.

Examples

Input	Output
3 3 1 3 1 3 1 3 1 2 2 3 3 1	-1

Input	Output
3 3 1 3 1 3 1 3 1 3 1 2 2 3 1 3	1 2 3

Pick up sticks is a fascinating game. A collection of coloured sticks are dumped in a tangled heap on the table. Players take turns trying to pick up a single stick at a time without moving any of the other sticks. It is very difficult to pick up a stick if there is another stick lying on top of it. The players therefore try to pick up the sticks in an order such that they never have to pick up a stick from underneath another stick.

Input

The input consists of several test cases. The first line of each test case contains two integers n and m each at least one and no greater than one million. The integer n is the number of sticks, and m is the number of lines that follow. The sticks are numbered from 1 to n . Each of the following lines contains a pair of integers a, b , indicating that there is a point where stick a lies on top of stick b . The last line of input is '0 0'. These zeros are not values of n and m , and should not be processed as such.

Output

For each test case, output n lines of integers, listing the sticks in the order in which they could be picked up without ever picking up a stick with another stick on top of it. If there are multiple such correct orders, any one will do. If there is no such correct order, output a single line containing the word 'IMPOSSIBLE'.

Sample Input

```
3 2
1 2
2 3
0 0
```

Sample Output

```
1
2
3
```



Problem V. Hit the Light Switches

Time limit 1000 ms

Mem limit 65536 kB

Ronju is a night-guard at the “Lavish office buildings Ltd.” headquarters. The office has a large grass field in front of the building. Every day, when Ronju comes to duty in the evening, it is his duty to turn on all the lights in the field. However, given the large size of the field and a large number of lights, it is very tiring for him to walk to each light to turn it on.

After a lot of thinking, he has devised an ingenious plan – he will swap the switches for light-sensitive triggers. A local electronic store nearby sells these funny trigger switches at a very cheap price. Once installed at a light-post, it will automatically turn that light on whenever it can sense some other light lighting up nearby. So, from now on, Ronju can just manually flip a few switches, and the light from those will trigger nearby sensors, which will in turn light up some more lights nearby, and so on, gradually lighting up the whole field.

Now Ronju wonders: how many switches does he have to flip manually for this?

Input

Input starts with an integer T (≤ 10), denoting the number of test cases.

Each case contains a blank line two integers N ($1 \leq N \leq 10000$) and M ($0 \leq M \leq 50000$), where N is the number of lights in the field, and M more lines of input follows in this input case. Each of these extra M lines will have two different integers a and b separated by a space, where $1 \leq a, b \leq N$, indicating that if the light a lights up, it will trigger the light b to turn on as well (according to their distance, brightness, sensor sensitivity, orientation and other complicated factors).

Output

For each case, print the case number and the minimum number of lights that Ronju must turn on manually before all the lights in the whole field gets lit up.

Sample

Input	Output
2	Case 1: 2
5 4	Case 2: 2
1 2	
1 3	
3 4	
5 3	
4 4	
1 2	
1 3	
4 2	
4 3	

Order is an important concept in mathematics and in computer science. For example, Zorns Lemma states: *a partially ordered set in which every chain has an upper bound contains a maximal element*. Order is also important in reasoning about the fix-point semantics of programs.

This problem involves neither Zorns Lemma nor fix-point semantics, but does involve order.

Given a list of variable constraints of the form $A < B$, you are to write a program that prints all orderings of the variables that are consistent with the constraints. For example, given the constraints $A < B$ and $A < C$ there are two orderings of the variables A , B , and C that are consistent with these constraints: ABC and ACB .

Input

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

The input consists of two lines: a list of variables on one line followed by a list of constraints of the form $A < B$ on the next line. Variables and constraints are separated by single spaces.

All variables are single character, upper-case letters. There will be at least two variables, and no more than 20 variables. There will be at least one constraint, and no more than 50 constraints. There will be no more than 300 orderings consistent with the constraints in a specification.

Output

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.

All orderings consistent with the constraints should be printed. Orderings are printed in alphabetical order, one per line. Characters on a line are separated by a space. If no ordering is possible the output is a single line with the word 'NO'.

Sample Input

```
1

A B F G
A<B B<F
```

Sample Output

```
A B F G
A B G F
A G B F
G A B F
```

Dilbert has just finished college and decided to go out with friends. Dilbert has some strange habits and thus he decided to celebrate this important moment of his life drinking a lot. He will start drinking beverages with low alcohol content, like beer, and then will move to a beverage that contains more alcohol, like wine, until there are no more available beverages. Once Dilbert starts to drink wine he will not drink beer again, so the alcohol content of the beverages never decreases with the time.

You should help Dilbert by indicating an order in which he can drink the beverages in the way he wishes.

Input

Each test case starts with $1 \leq N \leq 100$, the number of available beverages. Then will follow N lines, informing the name of each beverage, a name has less than 51 characters and has no white spaces. Then there is another line with an integer $0 \leq M \leq 200$ and M lines in the form $B_1 B_2$ will follow, indicating that beverage B_2 has more alcohol than beverage B_1 , so Dilbert should drink beverage B_1 before he starts drinking beverage B_2 . Be sure that this relation is transitive, so if there is also a relation $B_2 B_3$ you should drink B_1 before you can drink B_3 . There is a blank line after each test case. In the case there is no relation between two beverages Dilbert should start drinking the one that appears first in the input. The input is terminated by end of file (EOF).

Output

For each test case you must print the message: ‘**Case #C: Dilbert should drink beverages in this order:** $B_1 B_2 \dots B_N$.’, where C is the number of the test case, starting from 1, and $B_1 B_2 \dots B_N$ is a list of the beverages such that the alcohol content of beverage B_{i+1} is not less than the alcohol content of beverage B_{i-1} . After each test case you must print a blank line.

Sample Input

```
3
vodka
wine
beer
2
wine vodka
beer wine

5
wine
beer
rum
apple-juice
cachaca
6
beer cachaca
apple-juice beer
apple-juice rum
beer rum
beer wine
wine cachaca

10
cachaca
rum
apple-juice
tequila
whiskey
wine
vodka
beer
martini
gin
11
beer whiskey
apple-juice gin
rum cachaca
vodka tequila
apple-juice martini
rum gin
wine whiskey
apple-juice beer
beer rum
wine vodka
beer tequila
```

Sample Output

Case #1: Dilbert should drink beverages in this order: beer wine vodka.

Case #2: Dilbert should drink beverages in this order: apple-juice beer wine rum cachaca.

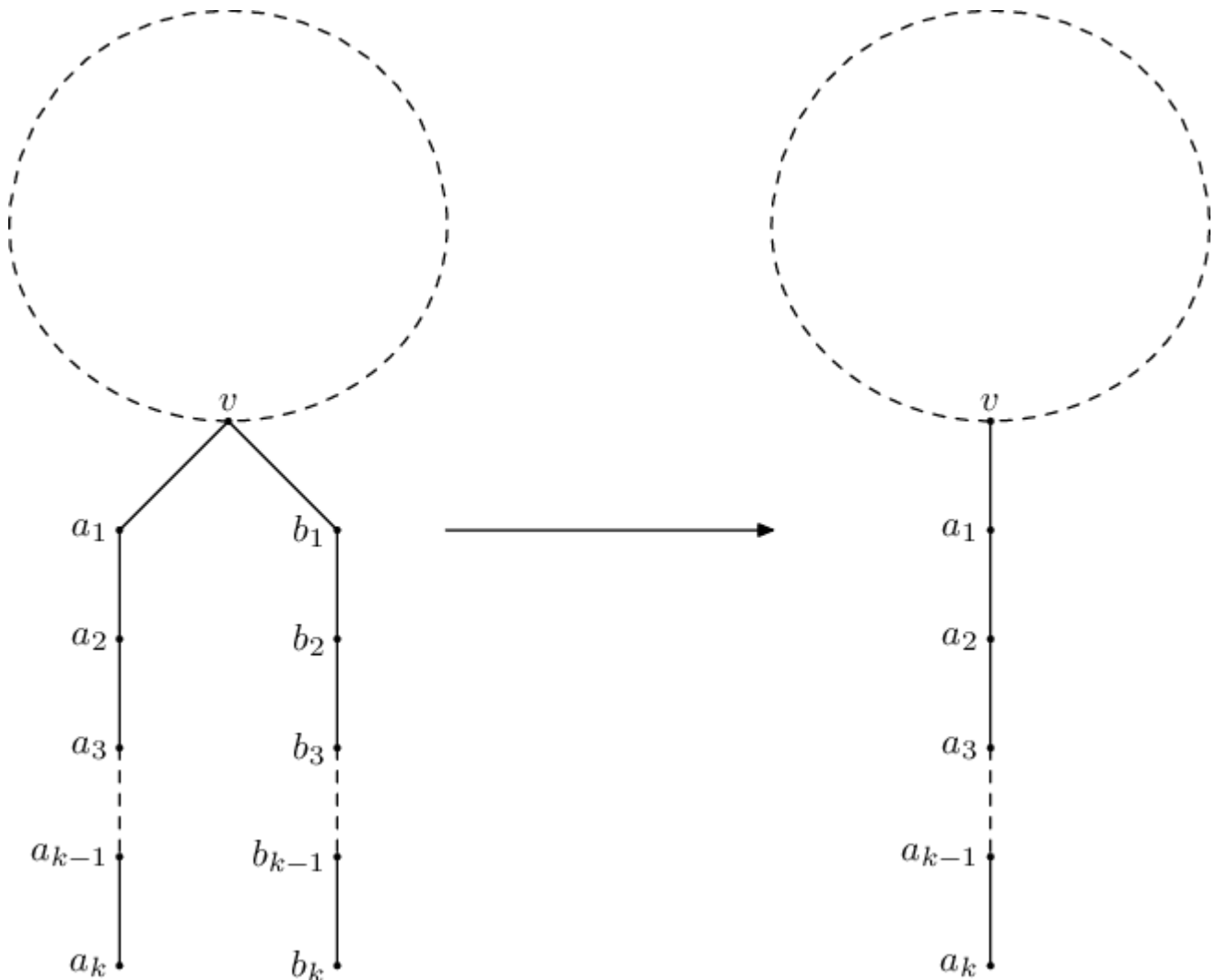
Case #3: Dilbert should drink beverages in this order: apple-juice wine vodka beer rum cachaca tequila whiskey martini gin.

Problem Y. Tree Folding

Time limit 2000 ms

Mem limit 524288 kB

Vanya wants to minimize a tree. He can perform the following operation multiple times: choose a vertex v , and two disjoint (except for v) paths of equal length $a_0 = v, a_1, \dots, a_k$, and $b_0 = v, b_1, \dots, b_k$. Additionally, vertices $a_1, \dots, a_k, b_1, \dots, b_k$ must not have any neighbours in the tree other than adjacent vertices of corresponding paths. After that, one of the paths may be merged into the other, that is, the vertices b_1, \dots, b_k can be effectively erased:



Help Vanya determine if it possible to make the tree into a path via a sequence of described operations, and if the answer is positive, also determine the shortest length of such path.

Input

The first line of input contains the number of vertices n ($2 \leq n \leq 2 \cdot 10^5$).

Next $n - 1$ lines describe edges of the tree. Each of these lines contains two space-separated integers u and v ($1 \leq u, v \leq n, u \neq v$) — indices of endpoints of the corresponding edge. It is guaranteed that the given graph is a tree.

Output

If it is impossible to obtain a path, print -1 . Otherwise, print the minimum number of edges in a possible path.

Examples

Input	Output
6 1 2 2 3 2 4 4 5 1 6	3

Input	Output
7 1 2 1 3 3 4 1 5 5 6 6 7	-1

Note

In the first sample case, a path of three edges is obtained after merging paths $2 - 1 - 6$ and $2 - 4 - 5$.

It is impossible to perform any operation in the second sample case. For example, it is impossible to merge paths $1 - 3 - 4$ and $1 - 5 - 6$, since vertex 6 additionally has a neighbour 7 that is not present in the corresponding path.

Problem Z. Online Courses In BSU

Time limit 2000 ms

Mem limit 262144 kB

Now you can take online courses in the Berland State University! Polycarp needs to pass k **main** online courses of his specialty to get a diploma. In total n courses are available for the passage.

The situation is complicated by the dependence of online courses, for each course there is a list of those that must be passed before starting this online course (the list can be empty, it means that there is no limitation).

Help Polycarp to pass the least number of courses in total to get the specialty (it means to pass all **main** and necessary courses). Write a program which prints the order of courses.

Polycarp passes courses consistently, he starts the next course when he finishes the previous one. Each course can't be passed more than once.

Input

The first line contains n and k ($1 \leq k \leq n \leq 10^5$) — the number of online-courses and the number of main courses of Polycarp's specialty.

The second line contains k distinct integers from 1 to n — numbers of main online-courses of Polycarp's specialty.

Then n lines follow, each of them describes the next course: the i -th of them corresponds to the course i . Each line starts from the integer t_i ($0 \leq t_i \leq n - 1$) — the number of courses on which the i -th depends. Then there follows the sequence of t_i distinct integers from 1 to n — numbers of courses in random order, on which the i -th depends. It is guaranteed that no course can depend on itself.

It is guaranteed that the sum of all values t_i doesn't exceed 10^5 .

Output

Print -1 , if there is no the way to get a specialty.

Otherwise, in the first line print the integer m — the minimum number of online-courses which it is necessary to pass to get a specialty. In the second line print m distinct integers — numbers of courses which it is necessary to pass in the chronological order of their passage. If there are several answers it is allowed to print any of them.

Examples

Input	Output
<pre> 6 2 5 3 0 0 0 2 2 1 1 4 1 5 </pre>	<pre> 5 1 2 3 4 5 </pre>

Input	Output
<pre> 9 3 3 9 5 0 0 3 9 4 5 0 0 1 8 1 6 1 2 2 1 2 </pre>	<pre> 6 1 2 9 4 5 3 </pre>

Input	Output
<pre> 3 3 1 2 3 1 2 1 3 1 1 </pre>	<pre> -1 </pre>

Note

In the first test firstly you can take courses number 1 and 2, after that you can take the course number 4, then you can take the course number 5, which is the main. After that you have to take only the course number 3, which is the last not passed main course.

Problem AA. Minimal Labels

Time limit 1000 ms

Mem limit 262144 kB

You are given a directed acyclic graph with n vertices and m edges. There are no self-loops or multiple edges between any pair of vertices. Graph can be disconnected.

You should assign labels to all vertices in such a way that:

- Labels form a valid permutation of length n — an integer sequence such that each integer from 1 to n appears exactly once in it.
- If there exists an edge from vertex v to vertex u then $label_v$ should be smaller than $label_u$.
- Permutation should be lexicographically smallest among all suitable.

Find such sequence of labels to satisfy all the conditions.

Input

The first line contains two integer numbers n, m ($2 \leq n \leq 10^5, 1 \leq m \leq 10^5$).

Next m lines contain two integer numbers v and u ($1 \leq v, u \leq n, v \neq u$) — edges of the graph. Edges are directed, graph doesn't contain loops or multiple edges.

Output

Print n numbers — lexicographically smallest correct permutation of labels of vertices.

Examples

Input	Output
3 3 1 2 1 3 3 2	1 3 2

Input	Output
4 5 3 1 4 1 2 3 3 4 2 4	4 1 2 3

Input	Output
5 4 3 1 2 1 2 3 4 5	3 1 2 4 5

Problem AB. Repairing Roads

Time limit	100 ms
Mem limit	1572864 kB
Code length Limit	50000 B
OS	Linux

The country of Byteland contains of N cities and $N - 1$ bidirectional roads between them such that there a path between any two cities. The roads in Byteland were built long ago and now they are in need of repair. You have been hired to repair all the roads. You intend to do this by dispatching robots on some of the roads. Each robot will repair the road he is currently on and then move to one of the adjacent unrepaired roads. After repairing that, he will move to another adjacent unrepaired road, repair that and so on. Two roads are adjacent if they have the same city at one of their endpoints. For the process to be efficient, no two robots will can ever repair the same road, and no road can be visited twice. What is the least number of robots needed to accomplish the task?

Input

The first line contains the number of test cases T . T test cases follow. The first line contains N , the number of cities in Byteland. The cities are numbered $0..N - 1$. The following $N - 1$ lines contain the description of the roads. The i th line contains two integers a_i and b_i , meaning that there is a road connecting cities with numbers a_i and b_i .

Output

Output T lines, one corresponding to each test case containing the required answer for that test case.

Constraints

$$1 \leq T \leq 20$$

$$1 \leq N \leq 10000$$

$$0 \leq a_i, b_i < N$$

Example

Input	Output
3 4 0 1 0 2 0 3 6 0 1 1 2 2 3 2 4 4 5 7 0 1 1 2 2 3 2 4 4 5 3 6	1 1 2

Explanation

For the first case, one robot is enough to repair all roads: $(0, 1) \rightarrow (0, 2) \rightarrow (0, 3)$

For the second case, one robot is again enough: $(0, 1) \rightarrow (1, 2) \rightarrow (2, 3) \rightarrow (2, 4) \rightarrow (4, 5)$

For the third case, there is no way to repair all the roads with one robot and at least two are needed.

Problem AC. Coins

Time limit 2000 ms

Mem limit 262144 kB

Input file `stdin`

Output file `stdout`

One day Vasya came across three Berland coins. They didn't have any numbers that's why Vasya didn't understand how their denominations differ. He supposed that if one coin is heavier than the other one, then it should be worth more. Vasya weighed all the three pairs of coins on pan balance scales and told you the results. Find out how the deminations of the coins differ or if Vasya has a mistake in the weighting results. No two coins are equal.

Input

The input data contains the results of all the weighting, one result on each line. It is guaranteed that every coin pair was weighted exactly once. Vasya labelled the coins with letters «A», «B» and «C». Each result is a line that appears as (letter)($>$ or $<$ sign)(letter). For example, if coin "A" proved lighter than coin "B", the result of the weighting is $A < B$.

Output

If the results are contradictory, print `Impossible`. Otherwise, print without spaces the rearrangement of letters «A», «B» and «C» which represent the coins in the increasing order of their weights.

Examples

Input	Output
A>B C<B A>C	CBA

Input	Output
A<B B>C C>A	ACB

Problem AD. Coprocessor

Time limit 1500 ms

Mem limit 262144 kB

You are given a program you want to execute as a set of tasks organized in a dependency graph. The dependency graph is a directed acyclic graph: each task can depend on results of one or several other tasks, and there are no directed circular dependencies between tasks. A task can only be executed if all tasks it depends on have already completed.

Some of the tasks in the graph can only be executed on a coprocessor, and the rest can only be executed on the main processor. In one coprocessor call you can send it a set of tasks which can only be executed on it. For each task of the set, all tasks on which it depends must be either already completed or be included in the set. The main processor starts the program execution and gets the results of tasks executed on the coprocessor automatically.

Find the minimal number of coprocessor calls which are necessary to execute the given program.

Input

The first line contains two space-separated integers N ($1 \leq N \leq 10^5$) — the total number of tasks given, and M ($0 \leq M \leq 10^5$) — the total number of dependencies between tasks.

The next line contains N space-separated integers $E_i \in \{0, 1\}$. If $E_i = 0$, task i can only be executed on the main processor, otherwise it can only be executed on the coprocessor.

The next M lines describe the dependencies between tasks. Each line contains two space-separated integers T_1 and T_2 and means that task T_1 depends on task T_2 ($T_1 \neq T_2$). Tasks are indexed from 0 to $N - 1$. All M pairs (T_1, T_2) are distinct. It is guaranteed that there are no circular dependencies between tasks.

Output

Output one line containing an integer — the minimal number of coprocessor calls necessary to execute the program.

Examples

Input	Output
<pre> 4 3 0 1 0 1 0 1 1 2 2 3 </pre>	2

Input	Output
<pre> 4 3 1 1 1 0 0 1 0 2 3 0 </pre>	1

Note

In the first test, tasks 1 and 3 can only be executed on the coprocessor. The dependency graph is linear, so the tasks must be executed in order $3 \rightarrow 2 \rightarrow 1 \rightarrow 0$. You have to call coprocessor twice: first you call it for task 3, then you execute task 2 on the main processor, then you call it for task 1, and finally you execute task 0 on the main processor.

In the second test, tasks 0, 1 and 2 can only be executed on the coprocessor. Tasks 1 and 2 have no dependencies, and task 0 depends on tasks 1 and 2, so all three tasks 0, 1 and 2 can be sent in one coprocessor call. After that task 3 is executed on the main processor.

Problem AE. Find Permutation

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

There is a length- N sequence $A = (A_1, \dots, A_N)$ that is a permutation of $1, \dots, N$.

While you do not know A , you know that $A_{X_i} < A_{Y_i}$ for M pairs of integers (X_i, Y_i) .

Can A be uniquely determined? If it is possible, find A .

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq M \leq 2 \times 10^5$
- $1 \leq X_i, Y_i \leq N$
- All values in the input are integers.
- There is an A consistent with the input.

Input

The input is given from Standard Input in the following format:

```
N M
X1 Y1
⋮
XM YM
```

Output

If A can be uniquely determined, print **Yes** in the first line. Then, print A_1, \dots, A_N in the second line, separated by spaces.

If A cannot be uniquely determined, just print **No**.

Sample 1

Input	Output
3 2 3 1 2 3	Yes 3 1 2

We can uniquely determine that $A = (3, 1, 2)$.

Sample 2

Input	Output
3 2 3 1 3 2	No

Two sequences $(2, 3, 1)$ and $(3, 2, 1)$ can be A .

Sample 3

Input	Output
4 6 1 2 1 2 2 3 2 3 3 4 3 4	Yes 1 2 3 4

Problem AF. Chat Screenshots

Time limit 2000 ms

Mem limit 262144 kB

There are n people in the programming contest chat. Chat participants are ordered by activity, but each person sees himself at the top of the list.

For example, there are 4 participants in the chat, and their order is $[2, 3, 1, 4]$. Then

- 1-st user sees the order $[1, 2, 3, 4]$.
- 2-nd user sees the order $[2, 3, 1, 4]$.
- 3-rd user sees the order $[3, 2, 1, 4]$.
- 4-th user sees the order $[4, 2, 3, 1]$.

k people posted screenshots in the chat, which show the order of participants shown to this user. The screenshots were taken within a short period of time, and the order of participants has not changed.

Your task is to determine whether there is a certain order that all screenshots correspond to.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of input test cases. The descriptions of test cases follow.

The first line of the description of each test case contains two integers n and k ($1 \leq k \leq n \leq 2 \cdot 10^5, n \cdot k \leq 2 \cdot 10^5$) — the number of chat participants and the number of participants who posted screenshots.

The following k lines contain descriptions of screenshots posted by the participants.

The i -th row contains n integers a_{ij} each ($1 \leq a_{ij} \leq n$, all a_{ij} are different) — the order of participants shown to the participant a_{i0} , where a_{i0} — the author of the screenshot. You can show that in the screenshot description it will always be at the top of the list.

It is guaranteed that the sum of $n \cdot k$ for all test cases does not exceed $2 \cdot 10^5$. It is also guaranteed that all the authors of the screenshots are different.

Output

Output t lines, each of which is the answer to the corresponding test case. As an answer, output "YES" if there exists at least one order of participants, under which all k screenshots could have been obtained. Otherwise, output "NO".

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Examples

Input	Output
10	YES
5 1	YES
1 2 3 4 5	YES
4 4	YES
1 2 3 4	NO
2 3 1 4	YES
3 2 1 4	YES
4 2 3 1	YES
6 2	YES
1 3 5 2 4 6	NO
6 3 5 2 1 4	
3 3	
1 2 3	
2 3 1	
3 2 1	
10 2	
1 2 3 4 5 6 7 8 9 10	
10 9 8 7 6 5 4 3 2 1	
1 1	
1	
5 2	
1 2 3 5 4	
2 1 3 5 4	
3 3	
3 1 2	
2 3 1	
1 3 2	
5 4	
3 5 1 4 2	
2 5 1 4 3	
1 5 4 3 2	
5 1 4 3 2	
3 3	
1 3 2	
2 1 3	
3 2 1	

Problem AG. Build Towers

Time limit	1500 ms
Code length Limit	50000 B
OS	Linux

There are N cities connected by roads such that they form a tree-like structure.

Your task is to build exactly A_i towers in i^{th} city. One or multiple towers can be built in a city only if there is a brick factory present in that city.

Currently, only city 1 has a brick factory. To complete the project, you can do the following two types of operations:

- Type 1: Construct a tower in a city that already has a brick factory.
- Type 2: Construct a brick factory in a city such that any of its neighbouring cities already have a brick factory.

Note that two cities are considered as neighbouring cities if there is a road connecting them.

Find the number of **different ways** to complete the project such that the total number of operations performed is **minimum**. Since the answer can be huge, print it modulo $10^9 + 7$.

Note: Two ways are considered different if the order and type of operations is not exactly the same for both ways.

Input Format

- The first line of input will contain a single integer T , denoting the number of test cases.
- Each test case consists of multiple lines of input.
 - The first line of each test case contains single integer N — the number of cities.
 - The second line of each test case contains N space-separated integers A_1, A_2, \dots, A_N denoting the number of towers to be built in i^{th} city.
 - The next $N - 1$ lines describe the roads. The i^{th} of these $N - 1$ lines contain two space-separated integers u_i and v_i , denoting a road between city u_i and city v_i .

Output Format

For each test case, output on a new line, the number of **different ways** to complete the project in **minimum cost** modulo $10^9 + 7$.

Two ways are considered different if the order and type of operations is not exactly the same for both ways.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq N \leq 2 \cdot 10^5$
- $0 \leq A_i \leq 10^6$
- $1 \leq u_i, v_i \leq N, u_i \neq v_i$
- The sum of A_i over any test case is at least 1 and won't exceed 10^6 .
- The sum of N over all test cases won't exceed $2 \cdot 10^5$.

Sample 1

Input	Output
3 4 0 0 1 0 1 2 2 3 2 4 3 1 1 0 1 2 2 3 3 1 0 1 1 2 1 3	1 3 3

- B_i represents construction of a brick factory in the city i . - T_i represents construction of a tower in the city i .

Test case 1: The only possible way to complete this project in minimum number of operations is $B_2B_3T_3$.

Test case 2: There are 3 possible ways to complete this project in minimum number of operations: $B_2T_2T_1$, $B_2T_1T_2$ and $T_1B_2T_2$.

Test case 3: There are 3 possible ways to complete this project in minimum number of operations: $B_3T_3T_1$, $B_3T_1T_3$ and $T_1B_3T_3$.

Problem AH. Permutation Counting

Time limit 5000 ms
Mem limit 524288 kB
OS Windows

You are given an integer n , and m pairs of integers (x_i, y_i) ($1 \leq i \leq m$) with distinct x_i , i.e. $x_i \neq x_j$ for all $i \neq j$. You need to find the number of permutation p of length n satisfying $p_{x_i} < p_{y_i}$ for all i . Since the number might be too large, you only need to output the answer modulo 998 244 353.

Recall that a permutation of length n is a sequence of length n and contains $1, 2, \dots, n$ exactly once. For instance, $\{1, 2, 3\}$ and $\{3, 1, 2\}$ are permutations while $\{1, 2, 4\}$ and $\{1, 2, 2\}$ are not.

Input

The first line contains two integers n ($1 \leq n \leq 2 \cdot 10^6$) and m ($0 \leq m \leq n$), indicating the length of the permutation and the number of constraints.

The i -th of the following m lines contains two integers x_i ($1 \leq x_i \leq n$) and y_i ($1 \leq y_i \leq n, x_i \neq y_i$) indicating the constraint that $p_{x_i} < p_{y_i}$.

Output

Output an integer indicating the number of permutations satisfying the conditions modulo 998 244 353.

Examples

Input	Output
3 1 1 2	3
Input	Output
3 2 1 2 2 3	1

Input	Output
3 2 1 2 2 1	0

Input	Output
10 7 1 2 3 2 5 4 7 8 2 6 4 6 8 6	37800

Note

In the first sample, all valid permutations are $\{1, 2, 3\}$, $\{1, 3, 2\}$ and $\{2, 3, 1\}$, so the answer is 3.

Problem A1. Triangles

Time limit 1000 ms
Mem limit 262144 kB
OS Windows

Walk_alone has a 500×500 square grid in a plane. The lower-left corner of the grid locates at $(0, 0)$, and all squares in it are of size 1×1 .

There are n segments in the grid, the i -th of which connecting point (x_i, y_i) and $(x_i - 1, y_i - 1)$. Now you are required to find out the number of triangles in the grid.

Input

The first line contains an integer n ($1 \leq n \leq 250\,000$), the number of segments in the grid.

The i -th of the next n lines contains two integers x_i and y_i ($1 \leq x_i, y_i \leq 500$) as described above. It is guaranteed that $(x_i, y_i) \neq (x_j, y_j)$ for all $i \neq j$.

Output

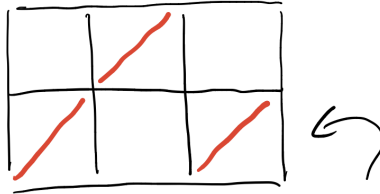
Output an integer representing the number of triangles.

Examples

Input	Output
3 1 1 2 2 3 1	8
Input	Output
6 1 1 2 1 3 2 4 3 1 3 2 4	20

Note

The figure below illustrates the first example.



Can you find
8 triangles ?

Problem AJ. Sorting It All Out

Time limit 1000 ms

Mem limit 10000 kB

An ascending sorted sequence of distinct values is one in which some form of a less-than operator is used to order the elements from smallest to largest. For example, the sorted sequence A, B, C, D implies that $A < B$, $B < C$ and $C < D$. In this problem, we will give you a set of relations of the form $A < B$ and ask you to determine whether a sorted order has been specified or not.

Input

Input consists of multiple problem instances. Each instance starts with a line containing two positive integers n and m . The first value indicates the number of objects to sort, where $2 \leq n \leq 26$. The objects to be sorted will be the first n characters of the uppercase alphabet. The second value m indicates the number of relations of the form $A < B$ which will be given in this problem instance. Next will be m lines, each containing one such relation consisting of three characters: an uppercase letter, the character "<" and a second uppercase letter. No letter will be outside the range of the first n letters of the alphabet. Values of $n = m = 0$ indicate end of input.

Output

For each problem instance, output consists of one line. This line should be one of the following three:

Sorted sequence determined after xxx relations: yyy...y.

Sorted sequence cannot be determined.

Inconsistency found after xxx relations.

where xxx is the number of relations processed at the time either a sorted sequence is determined or an inconsistency is found, whichever comes first, and yyy...y is the sorted, ascending sequence.

Sample

Input	Output
4 6 A<B A<C B<C C<D B<D A<B 3 2 A<B B<A 26 1 A<Z 0 0	Sorted sequence determined after 4 relations: ABCD. Inconsistency found after 2 relations. Sorted sequence cannot be determined.

In a certain city there are N intersections connected by one-way and two-way streets. It is a modern city, and several of the streets have tunnels or overpasses. Evidently it must be possible to travel between any two intersections. More precisely given two intersections V and W it must be possible to travel from V to W and from W to V .

Your task is to write a program that reads a description of the city street system and determines whether the requirement of connectedness is satisfied or not.

Input

The input contains several test cases. The first line of a test case contains two integers N and M , separated by a space, indicating the number of intersections ($2 \leq N \leq 2000$) and number of streets ($2 \leq M \leq N(N-1)/2$). The next M lines describe the city street system, with each line describing one street. A street description consists of three integers V , W and P , separated by a blank space, where V and W are distinct identifiers for intersections ($1 \leq V, W \leq N, V \neq W$) and P can be 1 or 2; if $P = 1$ the street is one-way, and traffic goes from V to W ; if $P = 2$ then the street is two-way and links V and W . A pair of intersections is connected by at most one street.

The last test case is followed by a line that contains only two zero numbers separated by a blank space.

Output

For each test case your program should print a single line containing an integer G , where G is equal to one if the condition of connectedness is satisfied, and G is zero otherwise.

Sample Input

```
4 5
1 2 1
1 3 2
2 4 1
3 4 1
4 1 2
3 2
1 2 2
1 3 2
3 2
1 2 2
1 3 1
4 2
1 2 2
3 4 2
0 0
```

Sample Output

```
1
1
0
0
```


Problem AL. Collecting Balls

Time limit 2000 ms

Mem limit 262144 kB

Problem Statement

There are $2N$ balls in the xy -plane. The coordinates of the i -th of them is (x_i, y_i) . Here, x_i and y_i are integers between 1 and N (inclusive) for all i , and no two balls occupy the same coordinates.

In order to collect these balls, Snuke prepared $2N$ robots, N of type A and N of type B. Then, he placed the type-A robots at coordinates $(1, 0), (2, 0), \dots, (N, 0)$, and the type-B robots at coordinates $(0, 1), (0, 2), \dots, (0, N)$, one at each position.

When activated, each type of robot will operate as follows.

- When a type-A robot is activated at coordinates $(a, 0)$, it will move to the position of the ball with the lowest y -coordinate among the balls on the line $x = a$, collect the ball and deactivate itself. If there is no such ball, it will just deactivate itself without doing anything.
- When a type-B robot is activated at coordinates $(0, b)$, it will move to the position of the ball with the lowest x -coordinate among the balls on the line $y = b$, collect the ball and deactivate itself. If there is no such ball, it will just deactivate itself without doing anything.

Once deactivated, a robot cannot be activated again. Also, while a robot is operating, no new robot can be activated until the operating robot is deactivated.

When Snuke was about to activate a robot, he noticed that he may fail to collect all the balls, depending on the order of activating the robots.

Among the $(2N)!$ possible orders of activating the robots, find the number of the ones such that all the balls can be collected, modulo 1 000 000 007.

Constraints

- $2 \leq N \leq 10^5$
- $1 \leq x_i \leq N$
- $1 \leq y_i \leq N$

- If $i \neq j$, either $x_i \neq x_j$ or $y_i \neq y_j$.

Inputs

Input is given from Standard Input in the following format:

```
N
x1 y1
...
x2N y2N
```

Outputs

Print the number of the orders of activating the robots such that all the balls can be collected, modulo 1 000 000 007.

Sample 1

Input	Output
2 1 1 1 2 2 1 2 2	8

We will refer to the robots placed at $(1, 0)$ and $(2, 0)$ as A1 and A2, respectively, and the robots placed at $(0, 1)$ and $(0, 2)$ as B1 and B2, respectively. There are eight orders of activation that satisfy the condition, as follows:

- A1, B1, A2, B2
- A1, B1, B2, A2
- A1, B2, B1, A2
- A2, B1, A1, B2
- B1, A1, B2, A2
- B1, A1, A2, B2
- B1, A2, A1, B2
- B2, A1, B1, A2

Thus, the output should be 8.

Sample 2

Input	Output
4 3 2 1 2 4 1 4 2 2 2 4 4 2 1 1 3	7392

Sample 3

Input	Output
4 1 1 2 2 3 3 4 4 1 2 2 1 3 4 4 3	4480

Sample 4

Input	Output
8 6 2 5 1 6 8 7 8 6 5 5 7 4 3 1 4 7 6 8 3 2 8 3 6 3 2 8 5 1 5 5 8	82060779

Sample 5

Input	Output
3 1 1 1 2 1 3 2 1 2 2 2 3	0

When there is no order of activation that satisfies the condition, the output should be 0.

Problem AM. The Queue

Time limit 1000 ms

Mem limit 65536 kB

On some special occasions Nadia's company provide very special lunch for all employees of the company. Before the food is served all of the employees must stand in a queue in front of the food counter. The company applied a rule for standing in the queue. The rule is nobody can stand anywhere in front of his supervisor in the queue. For example, if Abul is the supervisor of Babul and Abul stands in k^{th} position from the front of the queue, then Babul cannot stand at any position in between 1 and $k - 1$ from front of the queue.

The company has N employees and each of them has exactly one supervisor except one (CEO) who doesn't have any supervisor.

You have to calculate in how many ways the queue can be created. For this problem, you can safely assume that in at least one way the queue can be created.

Input

Input starts with an integer T (≤ 700), denoting the number of test cases.

Each case starts with a line containing an integer N ($1 \leq N \leq 1000$). Each of the following $N - 1$ lines will contain two integers a and b ($1 \leq a, b \leq N, a \neq b$), which denotes that a is the supervisor of b . For the sake of simplicity we are representing each employee by an integer number. Assume that the given input follows the restrictions stated above.

Output

For each case, print the case number and the number of ways to create the queue. The result can be large, print the result modulo 1000 000 007.

Sample

Input	Output
1 5 2 1 2 3 3 4 3 5	Case 1: 8

Problem AN. Connected Tree Subgraphs

Time limit 1000 ms
Mem limit 131072 kB
Code length Limit 10240 B

You are given a **tree** with N nodes. Count how many of the $N!$ permutations of the node labels lead to the tree respecting the following property:

- For any $K (1 \leq K \leq N)$ the **subgraph** induced by the nodes with labels $1, 2, \dots, K$ is **connected graph**" value="connected.

Standard input

The first line contains a single integer N .
 Each of the following $N - 1$ lines contains two integer values, representing two nodes that share an edge.

Standard output

Output a single integer representing the answer modulo $10^9 + 7$.

Constraints and notes

- $1 \leq N \leq 10^5$
- The nodes are numbered from 1 to N .

Example 1

Input	Output
3 1 2 2 3	4

Example 2

Input	Output
4 1 2 1 3 3 4	8

Example 3

Input	Output
7 1 2 1 3 2 4 2 5 3 6 3 7	240

Problem AO. Round Trip II

Time limit 1000 ms

Mem limit 524288 kB

Byteland has n cities and m flight connections. Your task is to design a round trip that begins in a city, goes through one or more other cities, and finally returns to the starting city. Every intermediate city on the route has to be distinct.

Input

The first input line has two integers n and m : the number of cities and flights. The cities are numbered $1, 2, \dots, n$.

Then, there are m lines describing the flights. Each line has two integers a and b : there is a flight connection from city a to city b . All connections are one-way flights from a city to another city.

Output

First print an integer k : the number of cities on the route. Then print k cities in the order they will be visited. You can print any valid solution.

If there are no solutions, print "IMPOSSIBLE".

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 2 \cdot 10^5$
- $1 \leq a, b \leq n$

Example

Input	Output
4 5 1 3 2 1 2 4 3 2 3 4	4 2 1 3 2

Problem AP. Fox And Names

Time limit 2000 ms

Mem limit 262144 kB

Input file `stdin`

Output file `stdout`

Fox Ciel is going to publish a paper on FOCS (Foxes Operated Computer Systems, pronounce: "Fox"). She heard a rumor: the authors list on the paper is always sorted in the lexicographical order.

After checking some examples, she found out that sometimes it wasn't true. On some papers authors' names weren't sorted in lexicographical order in normal sense. But it was always true that after some modification of the order of letters in alphabet, the order of authors becomes lexicographical!

She wants to know, if there exists an order of letters in Latin alphabet such that the names on the paper she is submitting are following in the lexicographical order. If so, you should find out any such order.

Lexicographical order is defined in following way. When we compare s and t , first we find the leftmost position with differing characters: $s_i \neq t_i$. If there is no such position (i. e. s is a prefix of t or vice versa) the shortest string is less. Otherwise, we compare characters s_i and t_i according to their order in alphabet.

Input

The first line contains an integer n ($1 \leq n \leq 100$): number of names.

Each of the following n lines contain one string $name_i$ ($1 \leq |name_i| \leq 100$), the i -th name.

Each name contains only lowercase Latin letters. All names are different.

Output

If there exists such order of letters that the given names are sorted lexicographically, output any such order as a permutation of characters 'a' – 'z' (i. e. first output the first letter of the modified alphabet, then the second, and so on).

Otherwise output a single word "Impossible" (without quotes).

Examples

Input	Output
3 rivest shamir adleman	bcdefghijklmnopqrstuvwxyz

Input	Output
10 tourist petr wjmzbr yeputons vepifanov scottwu oooooooooooooooooooo subscriber rowdark tankengineer	Impossible

Input	Output
10 petr egor endagorion feferivan ilovetanyaromanova kostka dmitriyh maratsnowbear bredorjaguarturnik cgyforever	aghjlnopefikdmbcqrstuvwxyz

Input	Output
7 car care careful carefully becarefuldontforgetsomething otherwiseyouwillbehacked goodluck	acdefghijklmnopqrstuvwxyz

Problem AQ. Constrained Topological Sort

Time limit 3000 ms

Mem limit 1048576 kB

Problem Statement

You are given a directed graph with N vertices and M edges. For $i = 1, 2, \dots, M$, the i -th edge is directed from vertex s_i to vertex t_i .

Determine whether there is a permutation $P = (P_1, P_2, \dots, P_N)$ of $(1, 2, \dots, N)$ that satisfies both of the following conditions, and if there is, provide an example.

- $P_{s_i} < P_{t_i}$ for all $i = 1, 2, \dots, M$.
- $L_i \leq P_i \leq R_i$ for all $i = 1, 2, \dots, N$.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $0 \leq M \leq \min\{4 \times 10^5, N(N-1)\}$
- $1 \leq s_i, t_i \leq N$
- $s_i \neq t_i$
- $i \neq j \implies (s_i, t_i) \neq (s_j, t_j)$
- $1 \leq L_i \leq R_i \leq N$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

```

N M
s1 t1
s2 t2
⋮
sM tM
L1 R1
L2 R2

```

\vdots
 $L_N \ R_N$

Output

If there is no P that satisfies the conditions, print **No**. If there is a P that satisfies the conditions, print **Yes** in the first line, and the elements of P separated by spaces in the second line, in the following format. If multiple P 's satisfy the conditions, any of them will be accepted.

Yes
 $P_1 \ P_2 \ \dots \ P_N$

Sample 1

Input	Output
5 4 1 5 2 1 2 5 4 3 1 5 1 3 3 4 1 3 4 5	Yes 3 1 4 2 5

$P = (3, 1, 4, 2, 5)$ satisfies the conditions. In fact,

- for the first edge $(s_1, t_1) = (1, 5)$, we have $P_1 = 3 < 5 = P_5$;
- for the second edge $(s_2, t_2) = (2, 1)$, we have $P_2 = 1 < 3 = P_1$;
- for the third edge $(s_3, t_3) = (2, 5)$, we have $P_2 = 1 < 5 = P_5$;
- for the fourth edge $(s_4, t_4) = (4, 3)$, we have $P_4 = 2 < 4 = P_3$.

Also,

- $L_1 = 1 \leq P_1 = 3 \leq R_1 = 5$,
- $L_2 = 1 \leq P_2 = 1 \leq R_2 = 3$,
- $L_3 = 3 \leq P_3 = 4 \leq R_3 = 4$,
- $L_4 = 1 \leq P_4 = 2 \leq R_4 = 3$,
- $L_5 = 4 \leq P_5 = 5 \leq R_5 = 5$.

Sample 2

Input	Output
2 2 1 2 2 1 1 2 1 2	No

No P satisfies the conditions, so print **No**.

Problem AR. Following Orders

Time limit 1000 ms

Mem limit 10000 kB

Order is an important concept in mathematics and in computer science. For example, Zorn's Lemma states: "a partially ordered set in which every chain has an upper bound contains a maximal element." Order is also important in reasoning about the fix-point semantics of programs.

This problem involves neither Zorn's Lemma nor fix-point semantics, but does involve order.

Given a list of variable constraints of the form $x < y$, you are to write a program that prints all orderings of the variables that are consistent with the constraints.

For example, given the constraints $x < y$ and $x < z$ there are two orderings of the variables x , y , and z that are consistent with these constraints: $x y z$ and $x z y$.

Input

The input consists of a sequence of constraint specifications. A specification consists of two lines: a list of variables on one line followed by a list of constraints on the next line. A constraint is given by a pair of variables, where $x y$ indicates that $x < y$.

All variables are single character, lower-case letters. There will be at least two variables, and no more than 20 variables in a specification. There will be at least one constraint, and no more than 50 constraints in a specification. There will be at least one, and no more than 300 orderings consistent with the constraints in a specification.

Input is terminated by end-of-file.

Output

For each constraint specification, all orderings consistent with the constraints should be printed. Orderings are printed in lexicographical (alphabetical) order, one per line.

Output for different constraint specifications is separated by a blank line.

Sample

Input	Output
a b f g a b b f v w x y z v y x v z v w v	abfg abgf agbf gabf wxzvy wzxvy xwzvy xzwvy zwxvy zxwvy

Problem AS. 排序

Time limit 1000 ms


Mem limit 128000 kB

Description



An ascending sorted sequence with distinct values refers to a sequence where elements increase from left to right. For example, an ordered sequence A, B, C, D represents $A < B, B < C, C < D$. In this problem, we will give you a series of relations in the form of $A < B$ and ask you to determine whether these relations can uniquely define the order of the sequence.


Input

The first line contains two positive integers n, m and n , where $2 \leq n \leq 26$ represents the number of elements to be sorted, and the 1-th to n -th elements are denoted by uppercase letters A, B, C, D, \dots . m indicates the number of given relations in the form of $A < B$.

The following m lines each contain 3 characters: a uppercase letter, a  symbol, and another uppercase letter, representing a relation between two elements.

Output

If the order of these n elements  (e.g., ) can be determined after the first x relations, output:

Sorted sequence determined after xxx relations: .

If a contradiction (e.g., $A < B, B < C, C < A$) is found after the first x relations, output:

Inconsistency found after x relations.

If the order of these n elements cannot be determined after all m relations, output:

Sorted sequence cannot be determined.

(Note: The program can terminate once the order of n elements is determined, and contradictions occurring afterward need not be considered.)

Sample 1

Input	Output
4 6 A<B A<C B<C C<D B<D A<B	Sorted sequence determined after 4 relations: ABCD.

Sample 2

Input	Output
3 2 A<B B<A	Inconsistency found after 2 relations.

Sample 3

Input	Output
26 1 A<Z	Sorted sequence cannot be determined.

Hint $2 \leq n \leq 26, 1 \leq m \leq 600.$

Problem AT. 游览

Time limit 1000 ms

Mem limit 128000 kB

Description

Having successfully passed Huang Yaoshi's test, you can now freely explore Peach Blossom Island!

You need to start from the west end of the island and play all the way to the east end, then leave from the dock at the east end. However, after your first visit, you find that the scenery of Peach Blossom Island is truly beautiful!!! So you want to take a boat from the dock at the east end back to the west end and play again, but there is a rule on Peach Blossom Island: you can explore as many times as you want, but each time the route you take cannot be exactly the same.

We abstract Peach Blossom Island as a graph, with n points representing intersections of paths, and m directed edges representing roads, which can only be traversed in the direction of the edge, and there may be multiple edges connecting the same two points. The input guarantees that this graph has no cycles and that there is at least one route from the west end to the east end. Two routes are considered different if and only if the paths they take are not completely identical.

Your task is to calculate how much time it will take to explore all the different routes.

Input

The first line contains 5 integers n, m, s, t, t_0 , representing the number of points, the number of edges, the identifier of the west end of the island, the identifier of the east end of the island (the identifiers range from 1 to n), and the time it takes to take the boat from the east end to the west end once.

The following m lines each contain 3 integers x, y, t , indicating that there is a path from point x to point y that takes t time to traverse.

Each line's multiple data points are separated by a space.

Output

Assuming the total time is $total$, output the value of $total \bmod 10000$ ($total$ modulo 10000).

Sample 1

Input	Output
3 4 1 3 7 1 2 5 2 3 7 2 3 10 1 3 15	56

Hint

【Sample Explanation】

There are a total of 3 paths from point 1 to point 3, which are $1 - 2 - 3$, $1 - 2 - 3$, and $1 - 3$.

The time calculation is:

$$(5 + 7) + 7 + (5 + 10) + 7 + (15) = 56$$

Data Range

$$2 \leq n \leq 10^4, 1 \leq m \leq 5 \times 10^4, t \leq 10^4, t_0 \leq 10^4.$$

Problem AU. 菜肴制作

Time limit 1000 ms

Mem limit 128000 kB

Description

Renowned food critic Mr. A has been invited to the ATM Grand Hotel to evaluate their dishes. The hotel has prepared n dishes for Mr. A, numbered from 1 to n in descending order of their estimated quality, with dish 1 being the highest-rated.

Due to flavor pairing considerations, certain dishes must be prepared before others. Specifically, there are m constraints in the form of “dish i must be prepared before dish j ,” which we abbreviate as (i, j) .

Now, the hotel hopes to determine an optimal dish preparation sequence that allows Mr. A to taste the highest-quality dishes as early as possible:

In other words,

1. Under all constraints, dish 1 should be prepared as early as possible.
2. Under all constraints and with dish 1 prioritized, dish 2 should be prepared as early as possible.
3. Under all constraints and with dishes 1 and 2 prioritized, dish 3 should be prepared as early as possible.
4. Under all constraints and with dishes 1, 2, and 3 prioritized, dish 4 should be prepared as early as possible.
5. And so on.

Example 1: With 4 dishes and two constraints $(3, 1)$ and $(4, 1)$, the preparation sequence is 3, 4, 1, 2.

Example 2: With 5 dishes and two constraints $(5, 2)$ and $(4, 3)$, the preparation sequence is 1, 5, 2, 4, 3.

In Example 1, first consider 1. Due to constraints $(3, 1)$ and $(4, 1)$, dish 3 and 4 must be prepared before 1. According to rule 3, dish 3 should also be prioritized over 4. Thus, the first three dishes in the sequence are determined as 3, 4, 1. Next, considering 2, the final sequence is determined as 3, 4, 1, 2.

In Example 2, preparing 1 first does not violate any constraints. Next, when considering 2, the constraint (5, 2) requires preparing 5 before 2. Then, when considering 3, the constraint (4, 3) requires preparing 4 before 3, resulting in the final sequence 1, 5, 2, 4, 3. Your task is to determine this optimal dish preparation sequence. If no solution exists, output **Impossible!** (with the first letter capitalized and the rest lowercase).

Input

The first line contains a positive integer t , indicating the number of test cases. This is followed by t test cases. For each test case:

The first line contains two space-separated positive integers n and m , representing the number of dishes and the number of preparation constraints, respectively.

The next m lines each contain two positive integers x, y , indicating that dish x must be prepared before dish y .

Output

The output file should contain t lines, each with n integers representing the optimal dish preparation sequence, or **Impossible!** if no solution exists.

Sample 1

Input	Output
3 5 4 5 4 5 3 4 2 3 2 3 3 1 2 2 3 3 1 5 2 5 2 4 3	1 5 3 4 2 Impossible! 1 5 2 4 3

Hint

【Sample Explanation】

In the second test case, the constraints require dish 1 to be prepared before dish 2, dish 2 before dish 3, and dish 3 before dish 1. This is impossible to satisfy, resulting in no solution.

【Data Range】

100% of the data satisfies $n, m \leq 10^5, 1 \leq t \leq 3$.

Note that among the m constraints, there may be identical constraints.

Problem AV. 车站分级

Time limit 1000 ms

Mem limit 131072 kB

Background

NOIP2013 Junior Group T4

Description

On a one-way railway line, there are n train stations numbered sequentially from $1, 2, \dots, n$. Each train station has a level, with the lowest being level 1. Several train services are operating on this line, each meeting the following requirement: if a train service stops at station x , then all stations between the starting station and the destination station with levels greater than or equal to station x must also be stopped at.

Note: the starting station and the destination station are naturally considered as known stations that need to be stopped at.

For example, the following table shows the operation of 5 train services. The first 4 train services all meet the requirement, while the 5th train service does not meet the requirement because it stops at station 3 (level 2) but does not stop at the passing station 6 (also level 2).

车站编号	1		2		3		4		5		6		7		8		9
车站级别	3		1		2		1		3		2		1		1		3
车次																	
1	始	→	→	→	停	→	→	→	停	→	终						
2					始	→	→	→	停	→	终						
3	始	→	→	→	→	→	→	→	停	→	→	→	→	→	→	→	终
4							始	→	停	→	停	→	停	→	停	→	终
5					始	→	→	→	停	→	→	→	→	→	→	→	终

Given the operation of m train services (all meeting the requirement), try to calculate the minimum number of different levels that these n stations can be divided into.

Input

The first line contains 2 positive integers n, m , separated by a space.

In the $i + 1$ th line ($1 \leq i \leq m$), there is initially a positive integer s_i ($2 \leq s_i \leq n$), indicating that the i th train service has s_i stopping stations; followed by s_i positive integers representing the numbers of all stopping stations in ascending order. There is a space between each pair of numbers. It is guaranteed that all train services meet the requirement.

Output

A positive integer, the minimum number of levels into which n stations can be divided.

Sample 1

Input	Output
9 2 4 1 3 5 6 3 3 5 6	2

Sample 2

Input	Output
9 3 4 1 3 5 6 3 3 5 6 3 1 5 9	3

Hint

For data of 20%, $1 \leq n, m \leq 10$;

For data of 50%, $1 \leq n, m \leq 100$;

For data of 100%, $1 \leq n, m \leq 1000$.

Problem AW. 神经网络

Time limit 1000 ms

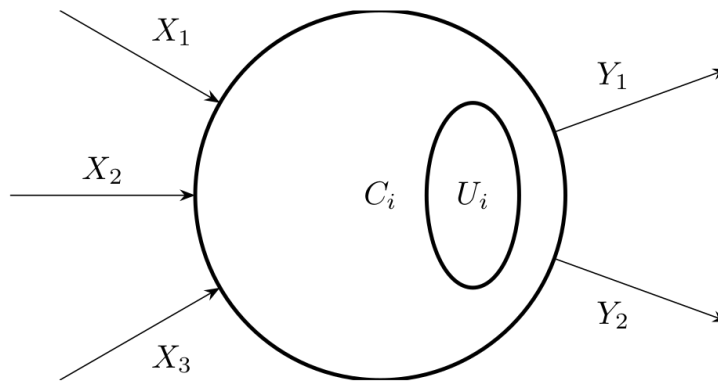
Mem limit 128000 kB

Background

Artificial Neural Networks (ANN) are emerging computational systems with self-learning capabilities, widely applied in pattern recognition, function approximation, loan risk assessment, and many other fields. Research on neural networks has always been a hot topic. After self-studying an introductory book on neural networks, Lanlan proposed a simplified model and hopes you can help validate its practicality through programming.

Description

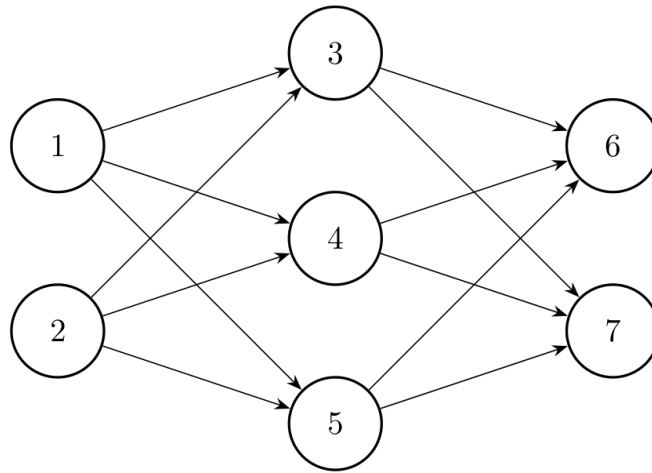
In Lanlan's model, a neural network is represented as a directed graph where nodes are called neurons, and there is at most one edge between any two neurons. The following is an example of a neuron:



Neuron (labeled as i)

In the diagram, $X_1 \sim X_3$ is the information input channel, $Y_1 \sim Y_2$ is the information output channel, C_i represents the current state of the neuron, and U_i is the threshold, which can be considered an intrinsic parameter of the neuron.

Neurons are arranged in a specific order to form the entire neural network. In Lanlan's model, neurons in the network are divided into several layers: the input layer, output layer, and several hidden layers. Each layer of neurons only sends information to the next layer and receives information from the previous layer. The following is an example of a simple three-layer neural network.



Lanlan stipulates that C_i follows the formula: (where n is the total number of neurons in the network)

$$C_i = \left(\sum_{(j,i) \in E} W_{ji} C_j \right) - U_i$$

In the formula, W_{ji} (which may be negative) represents the weight of the edge connecting neuron j and neuron i . When C_i is greater than 0, the neuron is in an excited state; otherwise, it remains calm. When a neuron is excited, it transmits signals to other neurons in the next second with a signal strength of C_i .

Thus, once the input layer neurons are activated, the entire network operates under the transmission of information. Now, given a neural network and the current state of the input layer neurons (C_i), your task is to compute the final state of the output layer.

Input

The first line of the input file contains two integers n ($1 \leq n \leq 100$) and p . The next n lines each contain 2 integers, where the $i + 1$ -th line represents the initial state and threshold (U_i) of neuron i . Non-input layer neurons initially have a state of 0. The following p lines each contain two integers i, j and an integer W_{ij} , indicating that the edge connecting neuron i, j has a weight of W_{ij} .

Output

The output file consists of several lines, each containing 2 integers: the neuron’s label and its final state, separated by spaces. Only output the states of output layer neurons whose final state is greater than 0, sorted in ascending order by label.

If all output layer neurons have a final state less than or equal to 0, output **NULL**.

Sample 1

Input	Output
5 6 1 0 1 0 0 1 0 1 0 1 1 3 1 1 4 1 1 5 1 2 3 1 2 4 1 2 5 1	3 1 4 1 5 1

Hint

【Problem Source】

NOIP 2003 Senior Group, Problem 1

Problem AX. 礼物

Time limit 2000 ms

Mem limit 512000 kB

Background

Since you've excellently completed the previous two problems, the kind-hearted `__stdcall` has decided to give you a small gift—a powerful boost for your journey towards solving this problem set.

Description

`__stdcall` decides to give you n gifts, each with a unique magic value a_i .

The magic values of these gifts are all distinct—no two are the same. These gifts possess special magic: if two gifts i, j have magic values satisfying $a_i \text{ bitand } a_j \geq \min(a_i, a_j)$, their magic will cancel each other out, so they cannot be placed in the same box.

Here, `bitand` represents the bitwise AND operation. If you're unfamiliar with this operation, please refer to: https://en.wikipedia.org/wiki/Bitwise_operation#AND.

ljt12138, the gift distributor, doesn't have many boxes, but fortunately, each box is large enough. Now, he asks for your help to allocate the gifts rationally, using the **minimum number of boxes** to contain all gifts. In other words, ensure each gift is placed in exactly one box, and no two gifts in the same box cancel each other's magic. If multiple valid solutions exist, you only need to provide **any one of them**.

ljt12138 is very kind—if you can only determine the required number of boxes, you can still earn 60% of the points for that test case. For details, refer to the hints and notes below.

Input

- The first line contains two numbers n and k , where n is the total number of gifts, and k is a parameter to facilitate your calculations.
- The second line contains n distinct numbers a_i , satisfying $0 \leq a_i < 2^k$, representing the magic values of the gifts.

Output

- The first line outputs a number. If you do not wish to provide a solution, output 0; if you want to output a solution, output 1. **If you output anything non-compliant here, it will be judged as WA.**
- The second line contains a number m , indicating that you've packed the gifts into m boxes.
- If you output 1 in the first line, the following m lines describe each box: first, a number s_i indicating the count of gifts in the current box, followed by s_i numbers representing the subset of gifts.

Sample 1

Input	Output
5 3 0 4 7 1 6	1 4 1 0 2 1 4 1 6 1 7

Hint

Additional Samples

You can download additional samples from

https://pan.baidu.com/s/1A8_ZA4yXXi5y6771x9JKUw.

About Outputting Solutions

- If you output 0 in the first line and correctly determine the minimum number of boxes required, you will earn 60% of the points for that test case.
- If you output 1 in the first line, correctly determine the minimum number of boxes, but fail to provide a valid solution, you will still earn 60% of the points.
- If you do not correctly determine the minimum number of boxes, you will earn no points for that test case.
- Contestants, please note: if your output does not follow the specified format, you will earn no points.

The relationship between data n , k is given in the table below:

Data ID	n	k
1	5	3
2	6	3
3	7	10
4	8	10
5	16	7
6	17	8
7	17	9
8	17	20
9	2×10^3	17
10	2.5×10^3	18
11	3×10^3	19
12	3×10^3	20
13	2.5×10^4	15
14	2.5×10^4	15
15	5×10^4	16
16	5×10^4	16
17	2.5×10^5	18
18	5×10^5	19
19	10^6	20
20	10^6	20

Problem AY. 灾难

Time limit 1000 ms

Mem limit 128000 kB

Background

Amoeba is Xiao Qiang's good friend.

Amoeba and Xiao Qiang were catching grasshoppers on the grassland. Suddenly, Xiao Qiang thought: if grasshoppers were hunted to extinction, the birds that feed on them would starve, and the raptors that prey on those birds would also go extinct, triggering a chain of ecological disasters.

Amoeba, who had studied biology, told Xiao Qiang that grasslands are extremely stable ecosystems. If grasshoppers went extinct, birds could still eat other insects, so the extinction of one species wouldn't necessarily cause major disasters.

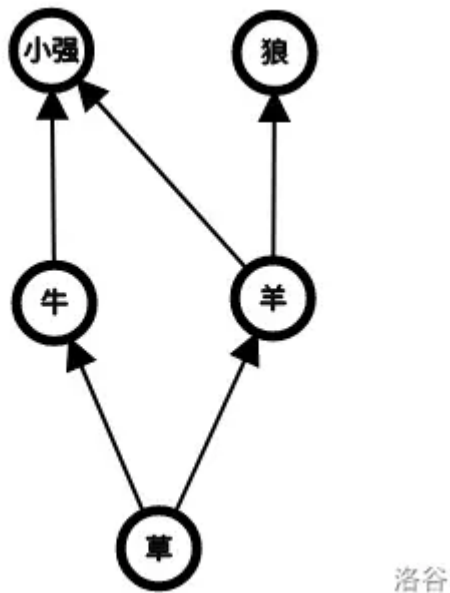
Description

Now let's examine this issue from a more professional perspective. We use a directed graph called a food web to describe biological relationships:

- A food web has n nodes, representing n species, numbered from 1 to n .
- If species x can eat species y , there is a directed edge from y to x .
- This graph contains no cycles.
- Some nodes have no outgoing edges—these represent producers that can survive through photosynthesis.
- Nodes with outgoing edges represent consumers that must feed on other species to survive.
- If all food sources for a consumer go extinct, it will also go extinct.

We define the “disaster value” of a species in the food web as the number of species that would go extinct if it suddenly disappeared.

For example: In a grassland, the biological relationships are as follows:



If Xiao Qiang and Amoeba scared all the sheep to death, the wolves would starve without food, while Xiao Qiang and Amoeba could survive by eating cows, and cows could survive by eating grass. Thus, the disaster value of sheep is 1. However, if grass suddenly went extinct, all 5 species on the grassland would perish, so the disaster value of grass is 4.

Given a food web, your task is to calculate the disaster value for each species.

Input

The first line contains an integer representing the number of nodes n in the food web.

From line 2 to line $(n + 1)$, each line contains several distinct integers. The integers $a_{i,j}$ in line $(i + 1)$ indicate that species i can eat species $a_{i,j}$. Each line ends with a 0 to mark its termination.

Output

Output n lines, each containing one integer. The i -th line should display the disaster value of species i .

Sample 1

Input	Output
5 0 1 0 1 0 2 3 0 2 0	4 1 0 0 0

Hint

Sample 1 Explanation

The sample input describes the example given in the problem statement.

Data Constraints

- For 50% of the data, it is guaranteed that $n \leq 10^4$.
- For 100% of the data, it is guaranteed that $1 \leq n \leq 65534$, $1 \leq a_{i,j} \leq n$, the input file size does not exceed 1 MB, and the graph contains no cycles.

Problem AZ. 杂务

Time limit 1000 ms

Mem limit 128000 kB

Description

John's farm has many chores to complete before milking the cows, and each chore requires a certain amount of time to finish. For example: they need to gather the cows, herd them into the barn, clean the cows' udders, and some other tasks. It is necessary to complete all chores as early as possible to have more time to extract more milk.

Of course, some chores can only be done after others are completed. For instance: you can only start cleaning the udders after herding the cows into the barn, and you cannot milk the cows before cleaning their udders. We call these tasks the prerequisites for completing the current task. There is at least one chore that does not require any prerequisites, which can be started earliest, marked as chore 1.

John has a list of n chores to complete, and this list has a certain order; the prerequisites for chore k ($k > 1$) can only be among chores 1 to $k - 1$.

Write a program to read in the description of each chore in order. Calculate the shortest time for all chores to be completed. Of course, unrelated chores can be worked on simultaneously, and you can assume that John's farm has enough workers to complete any number of tasks at the same time.

Input

Line 1: An integer n ($3 \leq n \leq 10,000$), the number of chores to be completed;

Lines 2 to $n + 1$: Each line contains some space-separated integers representing:

- Task number (guaranteed to be in ordered increasing from 1 to n in the input file);
- Time required to complete the task len ($1 \leq len \leq 100$);
- Some prerequisites that must be completed, with a total not exceeding 100 numbers, ending with a number 0. Some chores have no prerequisites and only describe a single 0.

It is guaranteed that there will be no extra spaces in the entire input file.

Output

An integer representing the shortest time required to complete all chores.

Sample 1

Input	Output
7 1 5 0 2 2 1 0 3 3 2 0 4 6 1 0 5 1 2 4 0 6 8 2 4 0 7 4 3 5 6 0	23

Problem BA. 最大食物链计数

Time limit 1000 ms

Mem limit 128000 kB

Background

Do you know about food chains? Delia got all the questions wrong about counting food chains in her biology exam because she always counted some chains multiple times or missed some. So she came to seek your help, but you don't know either! Write a program to help her out.

Description

You are given a food web, and you need to calculate the number of maximum food chains in this food web.

(The “maximum food chain” here refers to the **food chain in the biological sense**, where the **leftmost is a producer that does not prey on other organisms**, and the **rightmost is a consumer that is not preyed upon by other organisms**.)

Delia is in a hurry, so you only have 1 seconds.

Since this result may be too large, you only need to output the total count modulo 80112002.

Input

The first line contains two positive integers n 、 m , representing the number of species n and the number of predator-prey relationships m .

The next m lines each contain two positive integers, indicating that organism A is eaten by organism B.

Output

A single line containing an integer, which is the number of maximum food chains modulo 80112002.

Sample 1

Input	Output
5 7 1 2 1 3 2 3 3 5 2 5 4 5 3 4	5

Hint

Each test case satisfies the following conventions:

测试点编号	n	m
1,2	≤ 40	≤ 400
3,4	≤ 100	≤ 2000
5,6	≤ 1000	≤ 60000
7,8	≤ 2000	≤ 200000
9,10	≤ 5000	≤ 500000

【Additional Notes】

There will be no cycles in the data, satisfying biological requirements. (Thanks to [@AKEE](#))

Problem BB. Genealogical tree

Time limit 1000 ms

Mem limit 65536 kB

Special judge

The system of Martians' blood relations is confusing enough. Actually, Martians bud when they want and where they want. They gather together in different groups, so that a Martian can have one parent as well as ten. Nobody will be surprised by a hundred of children. Martians have got used to this and their style of life seems to them natural. And in the Planetary Council the confusing genealogical system leads to some embarrassment. There meet the worthiest of Martians, and therefore in order to offend nobody in all of the discussions it is used first to give the floor to the old Martians, than to the younger ones and only than to the most young childless assessors. However, the maintenance of this order really is not a trivial task. Not always Martian knows all of his parents (and there's nothing to tell about his grandparents!). But if by a mistake first speak a grandson and only than his young appearing great-grandfather, this is a real scandal. Your task is to write a program, which would define once and for all, an order that would guarantee that every member of the Council takes the floor earlier than each of his descendants.

Input

The first line of the standard input contains an only number N , $1 \leq N \leq 100$ — a number of members of the Martian Planetary Council. According to the centuries-old tradition members of the Council are enumerated with the natural numbers from 1 up to N . Further, there are exactly N lines, moreover, the I -th line contains a list of I -th member's children. The list of children is a sequence of serial numbers of children in a arbitrary order separated by spaces. The list of children may be empty. The list (even if it is empty) ends with 0.

Output

The standard output should contain in its only line a sequence of speakers' numbers, separated by spaces. If several sequences satisfy the conditions of the problem, you are to write to the standard output any of them. At least one such sequence always exists.

Sample

Input	Output
5 0 4 5 1 0 1 0 5 3 0 3 0	2 4 5 3 1

Problem BC. Window Pains

Time limit 1000 ms

Mem limit 65536 kB

Boudreaux likes to multitask, especially when it comes to using his computer. Never satisfied with just running one application at a time, he usually runs nine applications, each in its own window. Due to limited screen real estate, he overlaps these windows and brings whatever window he currently needs to work with to the foreground. If his screen were a 4 x 4 grid of squares, each of Boudreaux's windows would be represented by the following 2 x 2 windows:

<table><tr><td>1</td><td>1</td><td>.</td><td>.</td></tr><tr><td>1</td><td>1</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	1	1	.	.	1	1	<table><tr><td>.</td><td>2</td><td>2</td><td>.</td></tr><tr><td>.</td><td>2</td><td>2</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	.	2	2	.	.	2	2	<table><tr><td>.</td><td>.</td><td>3</td><td>3</td></tr><tr><td>.</td><td>.</td><td>3</td><td>3</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	.	.	3	3	.	.	3	3
1	1	.	.																																															
1	1	.	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	2	2	.																																															
.	2	2	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	.	3	3																																															
.	.	3	3																																															
.	.	.	.																																															
.	.	.	.																																															
<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>4</td><td>4</td><td>.</td><td>.</td></tr><tr><td>4</td><td>4</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	4	4	.	.	4	4	<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>5</td><td>5</td><td>.</td></tr><tr><td>.</td><td>5</td><td>5</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	5	5	.	.	5	5	<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>6</td><td>6</td></tr><tr><td>.</td><td>.</td><td>6</td><td>6</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr></table>	6	6	.	.	6	6
.	.	.	.																																															
4	4	.	.																																															
4	4	.	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	5	5	.																																															
.	5	5	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	.	6	6																																															
.	.	6	6																																															
.	.	.	.																																															
<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>7</td><td>7</td><td>.</td><td>.</td></tr><tr><td>7</td><td>7</td><td>.</td><td>.</td></tr></table>	7	7	.	.	7	7	.	.	<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>8</td><td>8</td><td>.</td></tr><tr><td>.</td><td>8</td><td>8</td><td>.</td></tr></table>	8	8	.	.	8	8	.	<table><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>.</td><td>.</td></tr><tr><td>.</td><td>.</td><td>9</td><td>9</td></tr><tr><td>.</td><td>.</td><td>9</td><td>9</td></tr></table>	9	9	.	.	9	9
.	.	.	.																																															
.	.	.	.																																															
7	7	.	.																																															
7	7	.	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	8	8	.																																															
.	8	8	.																																															
.	.	.	.																																															
.	.	.	.																																															
.	.	9	9																																															
.	.	9	9																																															

When Boudreaux brings a window to the foreground, all of its squares come to the top, overlapping any squares it shares with other windows. For example, if window 1 and then window 2 were brought to the foreground, the resulting representation would be:

1	2	2	?
1	2	2	?
?	?	?	?

If window 4 were then brought to the foreground:

1	2	2	?
4	4	2	?
4	4	?	?

?	?	?	?		?	?	?	?
---	---	---	---	--	---	---	---	---

... and so on ...

Unfortunately, Boudreaux's computer is very unreliable and crashes often. He could easily tell if a crash occurred by looking at the windows and seeing a graphical representation that should not occur if windows were being brought to the foreground correctly. And this is where you come in ...

Input

Input to this problem will consist of a (non-empty) series of up to 100 data sets. Each data set will be formatted according to the following description, and there will be no blank lines separating data sets.

A single data set has 3 components:

1. Start line - A single line:

START

2. Screen Shot - Four lines that represent the current graphical representation of the windows on Boudreaux's screen. Each position in this 4 x 4 matrix will represent the current piece of window showing in each square. To make input easier, the list of numbers on each line will be delimited by a single space.

3. End line - A single line:

END

After the last data set, there will be a single line:

ENDOFINPUT

Note that each piece of visible window will appear only in screen areas where the window could appear when brought to the front. For instance, a 1 can only appear in the top left quadrant.

Output

For each data set, there will be exactly one line of output. If there exists a sequence of bringing windows to the foreground that would result in the graphical representation of the windows on Boudreaux's screen, the output will be a single line with the statement:

THESE WINDOWS ARE CLEAN

Otherwise, the output will be a single line with the statement:
THESE WINDOWS ARE BROKEN

Sample

Input	Output
START 1 2 3 3 4 5 6 6 7 8 9 9 7 8 9 9 END START 1 1 3 3 4 1 3 3 7 7 9 9 7 7 9 9 END ENDOFINPUT	THESE WINDOWS ARE CLEAN THESE WINDOWS ARE BROKEN

Problem BD. Frame Stacking

Time limit 1000 ms

Mem limit 10000 kB

Consider the following 5 picture frames placed on an 9 x 8 array.

```

..... .CCC....

EEEEEE.. ..... ..BBBB.. .C.C....

E....E.. DDDDDD.. ..... ..B..B.. .C.C....

E....E.. D....D.. ..... ..B..B.. .CCC....

E....E.. D....D.. ....AAAA ..B..B.. .....

E....E.. D....D.. ....A..A ..BBBB.. .....

E....E.. DDDDDD.. ....A..A .....

E....E.. .....AAAA .....

EEEEEE.. .....

1         2         3         4         5

```

Now place them on top of one another starting with 1 at the bottom and ending up with 5 on top. If any part of a frame covers another it hides that part of the frame below.

Viewing the stack of 5 frames we see the following.

```

.CCC....

ECBCBB..

DCBCDB..

```

DCCC.B..

D.B.ABAA

D.BBBB.A

DDDDAD.A

E...AAAA

EEEEEE..

In what order are the frames stacked from bottom to top? The answer is EDABC.

Your problem is to determine the order in which the frames are stacked from bottom to top given a picture of the stacked frames. Here are the rules:

1. The width of the frame is always exactly 1 character and the sides are never shorter than 3 characters.
2. It is possible to see at least one part of each of the four sides of a frame. A corner shows two sides.
3. The frames will be lettered with capital letters, and no two frames will be assigned the same letter.

Input

Each input block contains the height, h ($h \leq 30$) on the first line and the width w ($w \leq 30$) on the second. A picture of the stacked frames is then given as h strings with w characters each.

Your input may contain multiple blocks of the format described above, without any blank lines in between. All blocks in the input must be processed sequentially.

Output

Write the solution to the standard output. Give the letters of the frames in the order they were stacked from bottom to top. If there are multiple possibilities for an ordering, list all such possibilities in alphabetical order, each one on a separate line. There will always be at least one legal ordering for each input block. List the output for all blocks in the input sequentially, without any blank lines (not even between blocks).

Sample

Input	Output
9 8 .CCC.... ECBCBB.. DCBCDB.. DCCC.B.. D.B.ABAA D.BBBB.A DDDDAD.A E...AAAA EEEEEE..	EDABC

Problem BE. 确定比赛名次

Time limit 1000 ms
Mem limit 32768 kB
OS Windows

有N个比赛队 ($1 \leq N \leq 500$)，编号依次为1，2，3，。。。。，N进行比赛，比赛结束后，裁判委员会要将所有参赛队伍从前往后依次排名，但现在裁判委员会不能直接获得每个队的比赛成绩，只知道每场比赛的结果，即P1赢P2，用P1，P2表示，排名时P1在P2之前。现在请你编程序确定排名。

Input

输入有若干组，每组中的第一行为二个数N ($1 \leq N \leq 500$)，M；其中N表示队伍的个数，M表示接着有M行的输入数据。接下来的M行数据中，每行也有两个整数P1，P2表示即P1队赢了P2队。

Output

给出一个符合要求的排名。输出时队伍号之间有空格，最后一名后面没有空格。

其他说明：符合条件的排名可能不是唯一的，此时要求输出时编号小的队伍在前；输入数据保证是正确的，即输入数据确保一定能有一个符合要求的排名。

Sample

Input	Output
4 3 1 2 2 3 4 3	1 2 4 3

Problem BF. Gym Class

Time limit 1000 ms

Mem limit 65536 kB

OS Windows

As we all know, Dudu Bear loves all kinds of sports activities.

Today, it finally became the long-awaited PE teacher. During the first class, it noticed something interesting. Before class begins, all students must line up in a single file. Initially, each student has a unique ID ranging from 1 to N . After forming the line, each student will determine the minimum ID among all students in front of them (including themselves) as their rating score for the class. The tricky part is that some students don't want certain other students to stand in front of them. Under this constraint, the newly appointed PE teacher—Dudu Bear—hopes to arrange the queue in a way that maximizes the sum of all students' rating scores.

Input

The first line contains an integer T , indicating T ($1 \leq T \leq 30$) test cases.

For each test case, the first line contains two integers N and M ($1 \leq N \leq 100000, 0 \leq M \leq 100000$), representing the total number of students and the number of specific student preferences, respectively.

The following M lines each contain two integers A and B ($1 \leq A, B \leq N$), indicating that the student with ID A does not want the student with ID B to stand before them. You may assume the input guarantees at least one valid arrangement exists.

Output

For each test case, output the maximum possible sum of scores.

Sample

Input	Output
3 1 0 2 1 1 2 3 1 3 1	1 2 6

Problem BG. Legal or Not

Time limit 1000 ms

Mem limit 32768 kB

OS Windows

ACM-DIY is a large QQ group where many excellent acmers get together. It is so harmonious that just like a big family. Every day, many "holy cows" like HH, hh, AC, ZT, lcc, BF, Qinz and so on chat on-line to exchange their ideas. When someone has questions, many warm-hearted cows like Lost will come to help. Then the one being helped will call Lost "master", and Lost will have a nice "prentice". By and by, there are many pairs of "master and prentice". But then problem occurs: there are too many masters and too many prentices, how can we know whether it is legal or not?

We all know a master can have many prentices and a prentice may have a lot of masters too, it's legal. Nevertheless, some cows are not so honest, they hold illegal relationship. Take HH and 3xian for instant, HH is 3xian's master and, at the same time, 3xian is HH's master, which is quite illegal! To avoid this, please help us to judge whether their relationship is legal or not.

Please note that the "master and prentice" relation is transitive. It means that if A is B's master and B is C's master, then A is C's master.

Input

The input consists of several test cases. For each case, the first line contains two integers, N (members to be tested) and M (relationships to be tested) ($2 \leq N$, $M \leq 100$). Then M lines follow, each contains a pair of (x, y) which means x is y's master and y is x's prentice. The input is terminated by N = 0.

TO MAKE IT SIMPLE, we give every one a number (0, 1, 2, ..., N-1). We use their numbers instead of their names.

Output

For each test case, print in one line the judgement of the messy relationship. If it is legal, output "YES", otherwise "NO".

Sample

Input	Output
3 2 0 1 1 2 2 2 0 1 1 0 0 0	YES NO

Problem BH. Reward

Time limit 1000 ms

Mem limit 32768 kB

OS Windows

Dandelion's uncle is a boss of a factory. As the spring festival is coming , he wants to distribute rewards to his workers. Now he has a trouble about how to distribute the rewards. The workers will compare their rewards ,and some one may have demands of the distributing of rewards ,just like a's reward should more than b's.Dandelion's unclue wants to fulfill all the demands, of course ,he wants to use the least money.Every work's reward will be at least 888 , because it's a lucky number.

Input

One line with two integers n and m ,stands for the number of works and the number of demands .($n \leq 10000$, $m \leq 20000$)

then m lines ,each line contains two integers a and b ,stands for a's reward should be more than b's.

Output

For every case ,print the least money dandelion 's uncle needs to distribute .If it's impossible to fulfill all the works' demands ,print -1.

Sample

Input	Output
2 1 1 2 2 2 1 2 2 1	1777 -1

Problem B1. 逃生

Time limit 1000 ms

Mem limit 32768 kB

OS Windows

A terrible incident has occurred, and now everyone is rushing to escape. However, the escape route is very narrow, forcing everyone to line up in a single row.

There are n people, numbered from 1 to n . Additionally, there are some peculiar constraints, each stating that person a must come before person b .

Moreover, society is unequal—some people are rich while others are poor. Person 1 is the wealthiest, person 2 is the second wealthiest, and so on. The wealthy individuals bribe the person in charge, giving them certain advantages.

The person in charge can now arrange the order of the queue. Since they've accepted bribes, they must ensure person 1 is as far forward as possible. If multiple arrangements still satisfy this, then person 2 should be as far forward as possible, and so on for person 3, etc.

Your task is to determine the order of the queue. We guarantee there is always a solution.

Input

The first line contains an integer T ($1 \leq T \leq 5$), representing the number of test cases.

For each test case, the first line contains two integers n ($1 \leq n \leq 30000$) and m ($1 \leq m \leq 100000$), indicating the number of people and the number of constraints, respectively.

Then, m lines follow, each containing two integers a and b , indicating a constraint that person a must come before person b . Note that a and b are always different.

Output

For each test case, output one line with the queue order, separated by spaces.

Sample

Input	Output
1 5 10 3 5 1 4 2 5 1 2 3 4 1 4 2 3 1 5 3 5 1 2	1 2 3 4 5

Problem BJ. Rank of Tetris

Time limit 1000 ms

Mem limit 32768 kB

OS Windows

Since Lele developed the Rating system, his Tetris career has soared to new heights, and soon he promoted the game worldwide.

To better cater to enthusiasts' preferences, Lele came up with another idea: he would create a global Tetris masters ranking list, updated regularly, aiming to outshine even the Forbes Billionaires List. As for the ranking criteria, it goes without saying that players would be sorted by Rating from highest to lowest. If two players share the same Rating, they would then be ranked by their RP (Reputation Points) from highest to lowest.

Finally, Lele is ready to take action, ranking N individuals. For convenience, each person has already been assigned a unique ID ranging from 0 to $N-1$, with higher IDs indicating higher RP.

Meanwhile, Lele obtained M pieces of Rating-related information from the paparazzi.

These pieces of information may fall into three categories: " $A > B$ ", " $A = B$ ", or " $A < B$ ", indicating that A's Rating is higher than, equal to, or lower than B's, respectively.

Now, Lele isn't asking you to compile the ranking list for him. He simply wants to know whether these pieces of information are sufficient to determine the ranking definitively. If so, output "OK". Otherwise, determine the reason for the ambiguity—whether it's due to incomplete information (output "UNCERTAIN") or conflicting information (output "CONFLICT").

Note: If the information contains both conflicts and incompleteness, output "CONFLICT".

Input

This problem contains multiple test cases. Process until the end of the file.

For each test case, the first line contains two integers N and M ($0 \leq N \leq 10000$, $0 \leq M \leq 20000$), representing the number of individuals to be ranked and the number of relations obtained, respectively.

The following M lines describe these relations.

Output

For each test case, output the result as specified in the problem statement in a single line.

Sample

Input	Output
3 3 0 > 1 1 < 2 0 > 2 4 4 1 = 2 1 > 3 2 > 0 0 > 1 3 3 1 > 0 1 > 2 2 < 1	OK CONFLICT UNCERTAIN

Problem BK. Sridhar Likes Travelling

Time limit	1000 ms
Code length Limit	50000 B
OS	Linux

Sridhar was a seasoned traveler. He liked to visit new places. More than all he was a meticulous planner. This time he was planning to visit Europe. He wrote down his travel itinerary like as follows:

If he wanted to visit Madrid, Paris, Munich, Warsaw and Kiev in this order, he would write it down like as:

```
Madrid Paris 100
Paris Munich 200
Munich Warsaw 150
Warsaw Kiev 120
```

More formally, if he wanted to go from **A** to **B** directly and the price is **C** dollars, then he would write

A B C

on a card. Each move was written on a different card. Sridhar was a great planner, so he would never visit the same place twice. Just before starting his journey, the cards got shuffled. Help Sridhar figure out the actual order of the cards and the total cost of his journey.

Input Format

The first line of the input contains an integer **T**, the number of test cases. **T** test cases follow. Each case contains an integer **N**, the number of cities Sridhar is planning to visit. **N-1** lines follow. Each line is of the form

A_i B_i C_i

where the **i-th** line refers to the **i-th** card after getting shuffled.

Output Format

For each case the output contains **N** lines

- The first $N-1$ lines should contain the $N-1$ cards in their proper original order
- The N -th line should contain the total cost of the travel in the format:

TotalCost : X \$

See Example for detailed format.

Constraints

$$1 \leq T \leq 10$$

$$1 \leq N \leq 5000$$

$$1 \leq \text{length of } A_i \leq 50$$

$$1 \leq \text{length of } B_i \leq 50$$

$$1 \leq C_i \leq 1000$$

A_i , B_i will contain only lowercase and uppercase latin characters, no two cities will have same names.

The names of cities are case-sensitive. So “warsaw” and “Warsaw” should be considered as different cities.

Sample 1

Input	Output
1 5 Warsaw Kiev 120 Madrid Paris 100 Munich Warsaw 150 Paris Munich 200	Madrid Paris 100 Paris Munich 200 Munich Warsaw 150 Warsaw Kiev 120 570\$

Problem BL. Present for Andrii

Time limit	1000 ms
Code length Limit	50000 B
OS	Linux

Read problems statements in [Mandarin Chinese](#) and [Russian](#) as well.

Andrii is a great programmer. He recently managed to enter Ukrainian IOI team. His parents wanted to appreciate him and present him a DAG(Directed Acyclic Graph). His favourite TV show is "Pimp My Ride" so he'd like to customize his graph. He'd like to add as many edges as possible in order to still have a DAG. Please, help Andrii, find the maximum number of edges he can add to his graph without obtaining any cycle.

Input

The first line of an input contains single integer N — the number of vertices in the graph. Vertices are numerated from 1 to N . Next N lines contain N characters each — adjacency matrix of the graph. If there is '1' in j^{th} character of the i^{th} line then there is an edge from vertex i to vertex j in Andrii's graph. It's guaranteed that given graph does not contain cycles.

Output

Output the maximal number of edges one can add to the graph in the first line. Then output the edges themselves. Edges should be written in separate lines, one by one. Edge is defined by a pair of integers a and b which means that edge from a to b should be added.

The author is pretty lazy and he does not want to write checking program. So, if there are multiple solutions which lead to maximal number of edges that can be added, then you should output lexicographically smallest sequence of edges. A sequence of edges is smaller than another if the first edge that they differ in is lexicographically smaller. Edges are compared as pairs of integers, i.e. as sequence of two integers.

Constraints

- $1 \leq N \leq 1500$

Example

Input:

3

0 1 0

0 0 0

0 0 0

Output:

2

1 3

2 3

Problem BM. Sort a String

Time limit 1000 ms

Code length Limit 50000 B

OS Linux

You are given a string S of length N .

You can apply the following operation on the string **any** number of times:

- Choose an index i ($1 \leq i < N$), and swap S_i with S_{i+1} , if S_i and S_{i+1} contain adjacent characters.

For example, **a** and **b** are adjacent, **z** and **y** are adjacent, **x** and **y** are adjacent, while **c** and **e** are not adjacent. Note that **a** and **z** are **not** considered adjacent.

Find the lexicographically **smallest** string you can obtain, by applying the above operation **any** number of times.

Note: String X is lexicographically smaller than string Y if $X_i < Y_i$, where i is the first index where X and Y differ.

Input Format

- The first line of input will contain a single integer T , denoting the number of test cases.
- Each test case consists of multiple lines of input.
 - The first line of each test case contains an integer N — the length of the string.
 - The second line contains the string S , consisting of lowercase english alphabets.

Output Format

For each test case, output on a new line – the lexicographically smallest string that we can obtain from the given string using any number of operations.

Constraints

- $1 \leq T \leq 10^4$
- $1 \leq N \leq 2 \cdot 10^5$

- S consists of lowercase english alphabets only.
- The sum of N over all test cases won't exceed $2 \cdot 10^5$.

Sample 1

Input	Output
2 5 cbaba 6 dbbced	bbcaa cdbbde

****Test case 1:**** For the given string `cbaba`:

- We swap **c** and **b** at indices 1 and 2 and the string becomes: **bcaba**.
- Swap **a** and **b** at indices 3 and 4 and the string becomes: **bcbaa**.
- Finally swap **c** and **b** at indices 2 and 3 and the string becomes: **bbcaa**.

Note that this is the lexicographically smallest string we can obtain using any number of operations.

Test case 2: For the given string **dbbced**:

- We swap **b** and **c** at indices 3 and 4 and the string becomes: **dbcbed**.
- Swap **b** and **c** at indices 2 and 3 and the string becomes: **dcbbbed**.
- Swap **d** and **c** at indices 1 and 2 and the string becomes: **cdbbbed**.
- Finally swap **e** and **d** at indices 5 and 6 and the string becomes: **cdbbde**.

Note that this is the lexicographically smallest string we can obtain using any number of operations.

Problem BN. Restore the Permutation

Time limit 1000 ms

Code length Limit 50000 B

OS Linux

Given a permutation P of size N , an array A of size N can be generated using the following method:

- If $P_i > P_j$ for all $(1 \leq j < i)$, set $A_i = 0$;
- Otherwise, set $A_i = \max(j)$ where $(1 \leq j < i)$ and $(P_j > P_i)$.

Unfortunately, we lost the permutation P and some elements of the array A .

The lost elements of the array A are denoted as $A_i = -1$. These elements can be replaced with any non-negative integer.

Given an array A with some lost elements, check whether there exists a permutation P , such that, we can obtain the array A from P using the above method.

If at least one such permutations exist, print the **lexicographically smallest** permutation.

If no such permutation exists, print -1 instead.

Note:

- A permutation of size N contains each element from 1 to N exactly once.
- Permutation X is said to be lexicographically smaller than permutation Y if:
 - $X_i < Y_i$;
 - $X_j = Y_j$ for all $(1 \leq j < i)$.

Input Format

- The first line of input will contain a single integer T , denoting the number of test cases.
- Each test case consists of multiple lines of input.
 - The first line of each test case contains an integer N — the size of A .
 - The next line contains N space-separated integers: A_1, A_2, \dots, A_N .

Output Format

For each test case, output on a new line:

- If at least one such permutations exist, print the **lexicographically smallest** permutation.
- If no such permutation exists, print -1 instead.

Constraints

- $1 \leq T \leq 10$
- $1 \leq N \leq 2 \cdot 10^5$
- $-1 \leq A_i < N$
- The sum of N over all test cases won't exceed $2 \cdot 10^5$.

Sample 1

Input	Output
4	1 4 3 2 5
5	-1
0 0 2 3 0	-1
5	1 2 3 4
0 0 1 2 4	
2	
1 -1	
4	
-1 -1 -1 -1	

****Test case1:**** There are 3 permutations which meet the condition

$\{[1, 4, 3, 2, 5], [2, 4, 3, 1, 5], [3, 4, 2, 1, 5]\}$. The lexicographically smallest permutation is $[1, 4, 3, 2, 5]$. The array A is obtained as: - A_1 : $A_1 = 0$ as there exists no $j < 1$ for which $P_i < P_j$. - A_2 : Since $P_2 > P_1$, $A_2 = 0$. - A_3 : We set A_3 as the maximum j such that $j < 3$ and $P_j > P_3$. Thus, $A_3 = j = 2$. - A_4 : We set A_4 as the maximum j such that $j < 4$ and $P_j > P_4$. Thus, $A_4 = j = 3$. - A_5 : Since P_5 is greater than P_1, P_2, P_3 , and P_4 , $A_5 = 0$.

Test case 2: No permutation meets the condition.

Test case 3: No permutation meets the condition.

Test case 4: Since all elements of the array are missing, we can assume the elements. Consider the array $[0, 0, 0, 0]$. The lexicographically smallest permutation using which the array A can be generated is $[1, 2, 3, 4]$.

Problem BO. Simple Sorting

Time limit 3047 ms
Code length Limit 50000 B
OS Linux

Given a list of numbers, you have to sort them in non decreasing order.

Input Format

- The first line contains a single integer, N , denoting the number of integers in the list.
- The next N lines contain a single integer each, denoting the elements of the list.

Output Format

Output N lines, containing one integer each, in non-decreasing order.

Constraints

- $1 \leq N \leq 10^6$
- $0 \leq \text{elements of the list} \leq 10^6$

Sample 1

Input	Output
5 5 3 6 7 1	1 3 5 6 7

Problem BP. Topological Sort

Time limit 2000 ms

Mem limit 1048576 kB

問題文

$(1, 2, \dots, N)$ の順列を以下では単に順列と呼びます。

正整数 N と順列 P が与えられます。

有向閉路と多重辺を持たない、頂点に 1 から N までの番号が付けられ、辺に番号が付けられていない N 頂点の有向グラフであって、辞書順最小トポロジカル順序が P と一致するようなものの個数を 998244353 で割った余りを求めてください。

▶ 辞書順最小トポロジカル順序の定義

制約

- $2 \leq N \leq 2 \times 10^5$
- (P_1, P_2, \dots, P_N) は $(1, 2, \dots, N)$ の順列
- 入力はすべて整数

入力

入力は以下の形式で標準入力から与えられる。

```
N
P_1 P_2 ... P_N
```

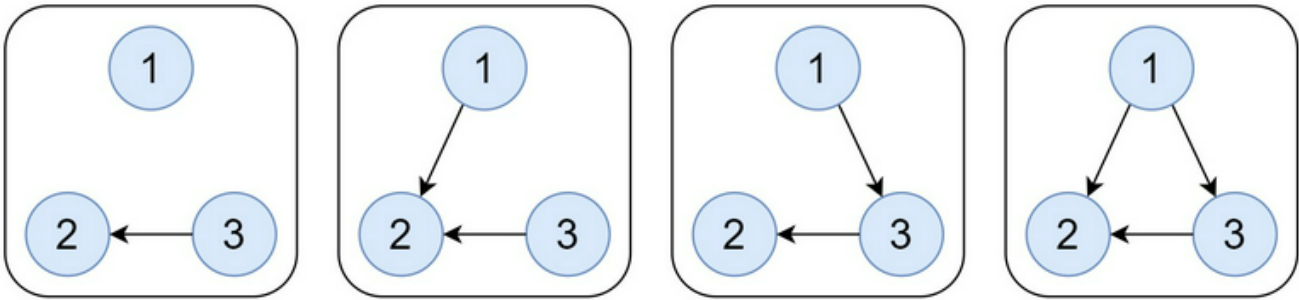
出力

答えを出力せよ。

Sample 1

Input	Output
3 1 3 2	4

以下の 4 つの有向グラフが条件を満たします。



Sample 2

Input	Output
5 1 2 3 4 5	1024

Sample 3

Input	Output
6 4 2 1 5 6 3	4096

Problem BQ. Restricted Permutation

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

Among the sequences P that are permutations of $(1, 2, \dots, N)$ and satisfy the condition below, find the lexicographically smallest sequence.

- For each $i = 1, \dots, M$, A_i appears earlier than B_i in P .

If there is no such P , print -1.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq M \leq 2 \times 10^5$
- $1 \leq A_i, B_i \leq N$
- $A_i \neq B_i$
- All values in input are integers.

Input

Input is given from Standard Input in the following format:

```
N M
A1 B1
⋮
AM BM
```

Output

Print the answer.

Sample 1

Input	Output
4 3 2 1 3 4 2 4	2 1 3 4

The following five permutations P satisfy the condition:

$(2, 1, 3, 4)$, $(2, 3, 1, 4)$, $(2, 3, 4, 1)$, $(3, 2, 1, 4)$, $(3, 2, 4, 1)$. The lexicographically smallest among them is $(2, 1, 3, 4)$.

Sample 2

Input	Output
2 3 1 2 1 2 2 1	-1

No permutations P satisfy the condition.

Problem BR. League

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

N players will participate in a tennis tournament. We will call them Player 1, Player 2, \dots , Player N .

The tournament is round-robin format, and there will be $N(N - 1)/2$ matches in total. Is it possible to schedule these matches so that all of the following conditions are satisfied? If the answer is yes, also find the minimum number of days required.

- Each player plays at most one matches in a day.
- Each player i ($1 \leq i \leq N$) plays one match against Player $A_{i,1}, A_{i,2}, \dots, A_{i,N-1}$ in this order.

Constraints

- $3 \leq N \leq 1000$
- $1 \leq A_{i,j} \leq N$
- $A_{i,j} \neq i$
- $A_{i,1}, A_{i,2}, \dots, A_{i,N-1}$ are all different.

Input

Input is given from Standard Input in the following format:

```
N
A1,1 A1,2 ... A1,N-1
A2,1 A2,2 ... A2,N-1
:
AN,1 AN,2 ... AN,N-1
```

Output

If it is possible to schedule all the matches so that all of the conditions are satisfied, print

the minimum number of days required; if it is impossible, print **-1**.

Sample 1

Input	Output
3 2 3 1 3 1 2	3

All the conditions can be satisfied if the matches are scheduled for three days as follows:

- Day 1: Player 1 vs Player 2
- Day 2: Player 1 vs Player 3
- Day 3: Player 2 vs Player 3

This is the minimum number of days required.

Sample 2

Input	Output
4 2 3 4 1 3 4 4 1 2 3 1 2	4

All the conditions can be satisfied if the matches are scheduled for four days as follows:

- Day 1: Player 1 vs Player 2, Player 3 vs Player 4
- Day 2: Player 1 vs Player 3
- Day 3: Player 1 vs Player 4, Player 2 vs Player 3
- Day 4: Player 2 vs Player 4

This is the minimum number of days required.

Sample 3

Input	Output
3 2 3 3 1 1 2	-1

Any scheduling of the matches violates some condition.

Problem BS. Longest Path

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

There is a directed graph G with N vertices and M edges. The vertices are numbered $1, 2, \dots, N$, and for each i ($1 \leq i \leq M$), the i -th directed edge goes from Vertex x_i to y_i . G **does not contain directed cycles**.

Find the length of the longest directed path in G . Here, the length of a directed path is the number of edges in it.

Constraints

- All values in input are integers.
- $2 \leq N \leq 10^5$
- $1 \leq M \leq 10^5$
- $1 \leq x_i, y_i \leq N$
- All pairs (x_i, y_i) are distinct.
- G **does not contain directed cycles**.

Input

Input is given from Standard Input in the following format:

```
 $N$   $M$ 
 $x_1$   $y_1$ 
 $x_2$   $y_2$ 
:
 $x_M$   $y_M$ 
```

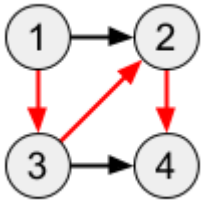
Output

Print the length of the longest directed path in G .

Sample 1

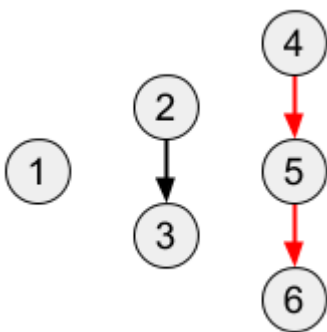
Input	Output
4 5 1 2 1 3 3 2 2 4 3 4	3

The red directed path in the following figure is the longest:

**Sample 2**

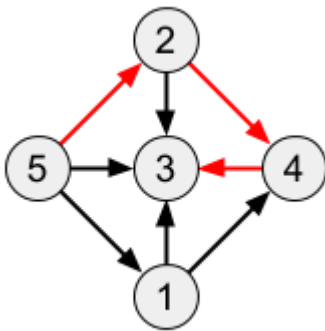
Input	Output
6 3 2 3 4 5 5 6	2

The red directed path in the following figure is the longest:

**Sample 3**

Input	Output
5 8 5 3 2 3 2 4 5 2 5 1 1 4 4 3 1 3	3

The red directed path in the following figure is one of the longest:



Problem BT. Make Adjacent

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

An integer sequence of length $2n$, $X = (X_1, X_2, \dots, X_{2n})$, such that $X_{2i-1} = X_{2i}$ for every $i = 1, 2, \dots, n$ is called a **good sequence**.

There is an integer sequence of length $2N$, $A = (A_1, A_2, \dots, A_{2N})$, which contains each integer $i = 1, 2, \dots, N$ exactly twice.

We want to make A a **good sequence** by performing the operation of swapping the values of two adjacent terms zero or more times.

Let K be the minimum number of operations we must perform the operation to make A a **good sequence**. Find the lexicographically smallest **good sequence** that can be obtained by performing the operations K times on A .

► What is lexicographical order on sequences?

Constraints

- $1 \leq N \leq 2 \times 10^5$
- $1 \leq A_i \leq N$
- A contains each integer $i = 1, 2, \dots, N$ exactly twice.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

```
N
A1 A2 ... A2N
```

Output

Print the lexicographically smallest **good sequence** that can be obtained by performing the operations K times on A , with spaces in between.

Sample 1

Input	Output
3 3 2 1 2 3 1	2 2 3 3 1 1

For example, we can perform the operation four times as $(3, 2, 1, 2, 3, 1) \rightarrow (3, 2, 1, 3, 2, 1) \rightarrow (3, 2, 3, 1, 2, 1) \rightarrow (3, 3, 2, 1, 2, 1) \rightarrow (3, 3, 2, 2, 1, 1)$ to make A a **good sequence**. This number is the minimum needed. With four operations, we can also make $A = (2, 2, 3, 3, 1, 1)$, and the answer is $(2, 2, 3, 3, 1, 1)$.

Sample 2

Input	Output
3 1 1 2 2 3 3	1 1 2 2 3 3

Sample 3

Input	Output
15 15 12 11 10 5 11 13 2 6 14 3 6 5 14 10 15 1 2 13 9 7 4 9 1 3 8 12 4 8 7	11 11 5 5 6 6 10 10 14 14 15 15 2 2 12 12 13 13 1 1 3 3 9 9 4 4 7 7 8 8

Problem BU. Typical Permutation Descriptor

Time limit 2000 ms

Mem limit 1048576 kB

Problem Statement

You are given a sequence of integers (A_1, \dots, A_N) of length N . This sequence satisfies $0 \leq A_i < i$ for each $i = 1, \dots, N$. Find the number of permutations (P_1, \dots, P_N) of $(1, \dots, N)$ that satisfy the following conditions, modulo 998244353.

- For each $i = 1, \dots, N$:
 - $P_j > P_i$ for any integer j with $A_i < j < i$
 - $P_{A_i} < P_i$ if $A_i > 0$

For the sequence (A_1, \dots, A_N) given in the input, it is guaranteed that there exists a permutation satisfying the conditions.

Constraints

- $1 \leq N \leq 3 \times 10^5$
- $0 \leq A_i < i$
- For A_1, \dots, A_N , there exists a permutation satisfying the conditions in the problem statement.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

```
N
A1 A2 ... AN
```

Output

Print the number of permutations satisfying the conditions, modulo 998244353.

Sample 1

Input	Output
4 0 1 0 3	3

There are three such permutations: (2, 3, 1, 4), (2, 4, 1, 3), and (3, 4, 1, 2).

Sample 2

Input	Output
22 0 1 2 2 2 2 2 2 1 9 9 9 9 0 14 15 15 15 14 19 19 19	353820794

The answer is 353820794, which is 2350309500 modulo 998244353.

Problem BV. SCC

Time limit 5000 ms

Mem limit 1048576 kB

Problem Statement

You are given a directed graph with N vertices and M edges, not necessarily simple. The i -th edge is oriented from the vertex a_i to the vertex b_i . Divide this graph into strongly connected components and print them in their topological order.

Constraints

- $1 \leq N \leq 500,000$
- $1 \leq M \leq 500,000$
- $0 \leq a_i, b_i < N$

Input

Input is given from Standard Input in the following format:

```
 $N$   $M$ 
 $a_0$   $b_0$ 
 $a_1$   $b_1$ 
 $\vdots$ 
 $a_{M-1}$   $b_{M-1}$ 
```

Output

Print $1 + K$ lines, where K is the number of strongly connected components. Print K on the first line. Print the information of each strongly connected component in next K lines in the following format, where l is the number of vertices in the strongly connected component and v_i is the index of the vertex in it.

```
 $l$   $v_0$   $v_1$   $\dots$   $v_{l-1}$ 
```

Here, for each edge (a_i, b_i) , b_i should not appear in **earlier** line than a_i .

If there are multiple correct output, print any of them.

Sample 1

Input	Output
6 7 1 4 5 2 3 0 5 5 4 1 0 3 4 2	4 1 5 2 4 1 1 2 2 3 0

Problem BW. Topological Sort

Time limit 2000 ms
Mem limit 131072 kB

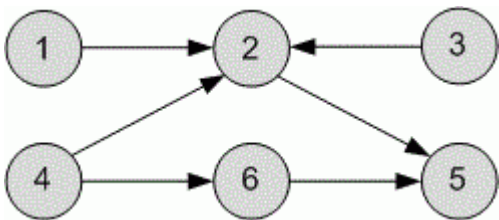
You are given a directed, unweighted graph. Perform a topological sort of its vertices.

Input

The first line contains two integers: the number of vertices n ($1 \leq n \leq 10^5$) and the number of edges m ($1 \leq m \leq 10^5$) in the graph. The following m lines describe the edges of the graph, each given by a pair of integers — the start and end vertex numbers.

Output

If a topological sort of the graph is possible, print any valid topological ordering of the vertices as a sequence of vertex numbers. If the graph cannot be topologically sorted, print -1 .



Examples

Input	Output
6 6 1 2 3 2 4 2 2 5 6 5 4 6	4 6 3 1 2 5

Problem BX. The Unique Topological Sort

Time limit 1000 ms
Mem limit 131072 kB

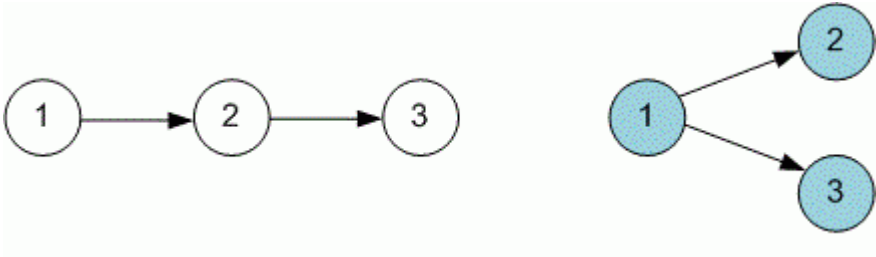
An unweighted directed graph is given. Determine whether it has a unique topological sort.

Input

The first line contains the number of vertices n ($1 \leq n \leq 2 \cdot 10^5$) and the number of edges m ($1 \leq m \leq 10^5$) in the graph. The following m lines describe the edges of the graph, each represented by a pair of integers — the indices of the start and end vertices.

Output

Print "YES" if there is a unique topological sort of the vertices, and "NO" if there are multiple valid sorts. If a topological sort is impossible, print -1 .



Examples

Input	Output
3 2 1 2 2 3	YES

Input	Output
3 2 1 2 1 3	NO

Problem BY. The smallest Topological Sort

Time limit 1000 ms

Mem limit 131072 kB

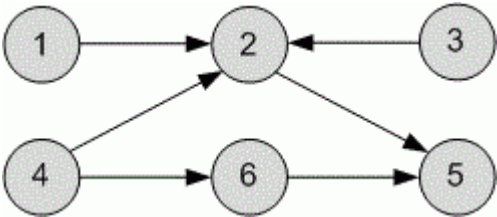
Given a directed unweighted graph. Find its lexicographically smallest topological sorting.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 10^5$) — the number of vertices and edges in the graph. The next m lines describe the edges of the graph. Each edge is given by a pair of integers — the starting and ending vertices.

Output

Print the lexicographically smallest topological sorting of the graph as a sequence of vertex numbers. If it is not possible to perform a topological sorting, print -1 .



Examples

Input	Output
6 6 1 2 3 2 4 2 2 5 6 5 4 6	1 3 4 2 6 5

Problem BZ. Inverse Topological Sort

Time limit 1000 ms

Mem limit 1048576 kB

OS Windows

Bobo recently learned the concept of topological ordering. One day, he observed a directed acyclic graph (DAG) $G = (V, E)$ with $|V| = n$ vertices numbered from 1 to n , and immediately wrote down on the paper two sequences $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$, such that A is the lexicographically smallest topological ordering of G , and B is the lexicographically largest topological ordering of G .

Unfortunately, now Bobo has forgotten what the original DAG G looks like, and all he has is the two sequences A and B . Can you help Bobo recover the original graph G ? There might be multiple possible graphs corresponding to A and B , or Bobo might write down the sequences incorrectly so that no valid graphs exist.

Refer to the note section for formal definitions of the underlined items.

Input

The first line of input contains an integer n ($1 \leq n \leq 10^5$), denoting the length of the two sequences.

The second line of input contains n **pairwise distinct** integers a_1, a_2, \dots, a_n ($1 \leq a_i \leq n$), denoting the first sequence A .

The second line of input contains n **pairwise distinct** integers b_1, b_2, \dots, b_n ($1 \leq b_i \leq n$), denoting the second sequence B .

Output

If there exists a directed graph G that satisfies the condition, output "Yes" in the first line; otherwise, output "No" in the first line. You can output each letter in any case (lowercase or uppercase). For example, the strings "yEs", "yes", "Yes", and "YES" will all be considered as positive replies.

If your answer is "Yes", output an integer m ($0 \leq m \leq \min(n(n-1)/2, 10^6)$) in the first line. Then, in the following m lines, output two integers u, v ($1 \leq u, v \leq n$) each, denoting

a directed edge (u, v) in the graph G . The graph G you output should satisfy that it is a directed acyclic graph, A is the lexicographically smallest topological ordering of G , and B is the lexicographically largest topological ordering of G . If multiple solutions exist, outputting any of them will be considered correct.

Note again that the graph you output must have no more than 10^6 edges. It can be shown that if there exists any valid graph, there exists a valid one with no more than 10^6 edges.

Examples

Input	Output
3 1 2 3 1 2 3	Yes 3 1 2 2 3 1 3

Input	Output
3 1 2 3 3 2 1	Yes 0

Input	Output
3 3 2 1 1 2 3	No

Note

Here, we provide formal definitions of some underlined items in the statement.

- A topological ordering of a directed graph $G = (V, E)$ is a linear ordering (i.e., permutation) of its vertices such that for every directed edge $(u, v) \in E$ from vertex u to vertex v , u comes before v in the ordering. It can be shown that a directed graph admits at least one topological ordering if and only if it is **acyclic**.
- For two sequences $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ with the same length n , A is said to be lexicographically smaller than B if and only if there exists some index $1 \leq i \leq n$, such that
 - $a_i < b_i$;
 - $a_j = b_j$ for all $j < i$.