The 2021 icpc

Asia Macau Regional Contest

Contest Session

April 3, 2022

澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

科技學院
Faculdade de Ciências e Tecnologia
Faculty of Science and Technology

香港中文大學
The Chinese University of Hong Kong

Problem List

A	So I'll Max Out My Constructive Algorithm Skills
B	The Matching System
C	Laser Trap
D	Shortest Path Fast Algorithm
E	Pass the Ball!
F	Sandpile on Clique
G	Cyclic Buffer
H	Permutation on Tree
I	LCS Spanning Tree
J	Colorful Tree
K	Link-Cut Tree

This problem set should contain 11 (eleven) problems on 14 (fourteen) numbered pages. Please inform a runner immediately if something is missing from your problem set.

Prepared by SUA Programming Contest Problem Setter Team.
https://sua.ac/

Hosted by

澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

科技學院
Faculdade de Ciências e Tecnologia
Faculty of Science and Technology

香港中文大學
The Chinese University of Hong Kong

Problem Set Prepared by

It's against the rules to open non-contest websites during the contest.
If you're interested (which is our pleasure),
please scan the QR code only after the contest.

Problem A. So I'll Max Out My Constructive Algorithm Skills

BaoBao the Witch is stuck in a maze with n rows and n columns, where the height of the cell in the i-th row and the j-th column is $h_{i,j}$. To get out of the maze, BaoBao has to find a path which passes through each cell exactly once. Each time she can only move into the neighboring cell sharing a same edge with the current one. But as we know, BaoBao is super lazy, so every time when she climbs up (that is to say, moving from a cell with a smaller height to another with a larger height) her happiness value will decrease. As her helping hand, your task is to find a valid path so that when moving along the path, the number of times BaoBao climbs up will not be more than the number of times she climbs down.

More formally, you need to find a sequence $(x_1, y_1), (x_2, y_2), \cdots, (x_{n^2}, y_{n^2})$ such that:

- For all $1 \le i \le n^2$, $1 \le x_i, y_i \le n$;
- For all $1 \le i, j \le n^2, i \ne j$, $(x_i, y_i) \ne (x_j, y_j)$;
- For all $2 \le i \le n^2$, $|x_i - x_{i-1}| + |y_i - y_{i-1}| = 1$;
- $\sum_{i=2}^{n^2} [h_{x_{i-1},y_{i-1}} < h_{x_i,y_i}] \le \sum_{i=2}^{n^2} [h_{x_{i-1},y_{i-1}} > h_{x_i,y_i}]$, where $[P]$ equals 1 when P is true, and equals 0 when it is false.

Additionally, you discover that the heights in all cells are a permutation of n^2, so you just need to output the height of each cell in a valid path.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 100$) indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 64$) indicating the size of the maze.

For the following n lines, the i-th line contains n integers $h_{i,1}, h_{i,2}, \cdots, h_{i,n}$ ($1 \le h_{i,j} \le n^2$) where $h_{i,j}$ indicates the height of the cell in the i-th row and the j-th column. It's guaranteed that all integers in the input make up a permutation of n^2.

Output

For each test case output one line containing n^2 separated by a space indicating the heights of each cell in a valid path. If there are multiple valid answers you can output any of them. It's easy to prove that an answer always exists.

Please, DO NOT output extra spaces at the end of each line, or your answer may be considered incorrect!

Example

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

Problem B. The Matching System

As the leader of the safety department, you are asked to check an ancient matching system in your company.

The system is fed with two strings, a to-be-matched string and a pattern string, and will determine whether the former can match the latter. The former string is a strict binary string(i.e., contains only **0** and **1**), and the latter string consists of four types of characters **0**, **1**, *, ^, where * means match zero or more arbitrary binary characters, and ^ means match exactly one binary character.

The system has two matching methods: maximum matching and minimum matching.

Consider the starting positions of the two strings. The maximum matching method will make different decisions based on the current character of the pattern string:

- *: The system will enumerate i from L to 0, where L is the remaining length of the to-be-matched string. Before each enumeration starts, the system consumes 1 unit of energy. Then it temporarily assumes that the current * in the pattern string matches the consecutive i characters in the to-be-matched string, and tries to match the remaining positions of two strings recursively. As long as one attempt is successful, the system will give up the remaining enumeration and stop the whole system. Otherwise, it will try the next enumeration until all attempts are tried and finally return to the previous * enumeration.
- **0,1**: The system will stop and return to the previous * enumeration if the to-be-matched string has been exhausted. Otherwise, it consumes 1 unit of energy to compare the current characters between the pattern string and the to-be-matched string. It will continue analyzing the remaining positions of these two strings if the result is the same, otherwise, return back to the previous * enumeration.
- ^: The system will stop and return to the previous * enumeration if the to-be-matched string has been exhausted. Otherwise, it consumes 1 unit of energy and moves on of two strings.

When the pattern string is exhausted, the system will check the to-be-matched string at the same time. It will return"Yes" and stop the whole process if the to-be-matched string is also exhausted, otherwise, it will return to the previous * enumeration. After all attempts are tried and no matching method is found, the system will eventually return "No".

Minimum matching does a similar thing except for the enumeration order of * (i.e., enumerate i from 0 to L).

These two matching methods seem not very effective, so you want to hack them. Please construct both a pattern string and a to-be-matched string of length n for each matching method, so that the system answers "Yes" and the energy consumption is as large as possible.

Input

There is only one test case in each test file.

The first and only line contains an integer n ($1 \leq n \leq 10^3$) indicating the length of strings need to be constructed.

Output

Please output the pattern string, the to-be-matched string, and the energy cost for the maximum matching method in the first 3 lines. Then output the pattern string, the to-be-matched string, and the energy cost for the minimum matching method in the next 3 lines.

If there are multiple constructing ways, you can output any of them.

The energy cost may be very large, so you need to output the value modulo $(10^9 + 7)$. Note that this is only for your convenience and you need to maximize the energy cost before the modulus.

Example

standard input	standard output
3	*0* 011 8 **1 101 7

Problem C. Laser Trap

BaoBao is playing the famous game *Elden Ring* these days. It's an open-world game in which you can control your character to travel from places to places. However, your character could also enter a trap and you need to figure out how to escape. Right now, BaoBao's character is stuck in a 2-dimensional plane with deadly lasers. There are n laser generators (each can be regarded as a point) shooting laser beams between every pair of them (so there are $\frac{n(n-1)}{2}$ laser beams in total). The beams start and end at generator points and do not stretch to infinity.

Starting at point $(0, 0)$, BaoBao wants to escape to point $(10^{10^{10^{10^{10}}}}, 10^{10^{10^{10^{10}}}})$ without touching any laser beam or generator. In order to do so, BaoBao can ask her friend DreamGrid to remove any number of laser generators, together with any laser beam that starts or ends at these generators. Output the minimum number of laser generators that need to be erased for the escape.

Note that BaoBao does not need to move in a specific direction to escape. Her escaping route can even be a curve if necessary.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^6$) indicating the number of laser generators.

For the following n lines, the i-th line contains two integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) indicating the location of the i-th laser generator.

It is guaranteed that no two generators coincide, and no laser beam or generator will touch $(0, 0)$.

It is also guaranteed that the sum of n of all test cases will not exceed 10^6.

Output

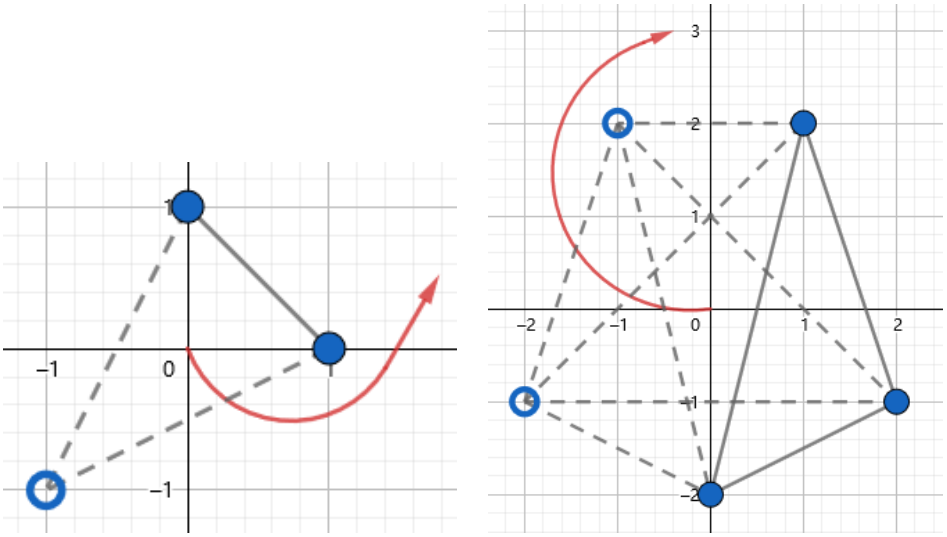
For each test case output one line containing one integer indicating the minimum number of generators that need to be removed.

Example

standard input	standard output
3 2 1 0 2 0 3 1 0 0 1 -1 -1 5 2 -1 1 2 -1 2 -2 -1 0 -2	0 1 2

Note

The second and the third sample test cases are shown below. Solid dots and lines represent the remaining laser generators and beams, while hollow dots and dashed lines represent the removed laser generators and beams. The arrow is the escaping route.



Problem D. Shortest Path Fast Algorithm

Recently, BaoBao has learned the Shortest Path Fast Algorithm (SPFA, or more formally, Bellman-Ford-Moore Algorithm) to solve the shortest path problem efficiently. He realizes that the algorithm looks so similar to the Dijkstra's algorithm after replacing the FIFO queue with priority queue, and shows you the below pseudo code.

Algorithm 1 The Shortest Path Fast Algorithm

```
1: function SPFA(G, s)                          ▷ G is the given graph and s is the source vertex
2:     Create vertex priority queue Q
3:     for each vertex v in G do
4:         dist[v] ← +∞                          ▷ Unknown distance from s to v
5:         vis[v] ← false                        ▷ No vertex is in Q at the beginning
6:     end for
7:     dist[s] ← 0                               ▷ Initialize distance from s to s to be 0
8:     Add s into Q with priority value 0
9:     vis[s] ← true
10:    cnt ← 0                                   ▷ Number of times we poll the priority queue
11:    while Q is not empty do
12:        Pick and remove best vertex u from Q  ▷ Explained below
13:        vis[u] ← false
14:        cnt ← cnt + 1
15:        for each neighbor v of u in G do
16:            d ← dist[u] + w_{u,v}             ▷ w_{u,v} is the weight of edge connecting vertices u and v in G
17:            if dist[v] > d then
18:                dist[v] ← d                   ▷ Update dist[v] if d is better
19:                if vis[v] is false then
20:                    Add v into Q with priority value d
21:                    vis[v] ← true
22:                end if
23:            end if
24:        end for
25:    end while
26: end function
```

By picking the best vertex from Q we mean picking the vertex with the smallest priority value (in case that multiple vertices have the smallest priority value, pick the vertex with the largest index among them).

You, the future computer scientist, find the BaoBao-modified SPFA algorithm works so slow in some carefully construted graph. However, BaoBao is sure that his algorithm works well, unless you show him a simple undirected graph that makes the variable `cnt` in the SPFA function no less than a certain k **at some time**.

Just teach him a lesson!

Input

There is only one test case in each test file.

The first and only line of the input contains a single integer k where $k = 1$ for the sample test case and $k = 10^5$ for the only secret test case.

Output

Output several lines in the following format to describe the input data of a simple undirected graph that makes the variable `cnt` in the SPFA function no less than k **at some time**.

The first line contains two integers n ($1 \le n \le 100$) and m ($0 \le m \le 10^3$), indicating the number of vertices and edges in the graph.

Then m lines follow, the i-th of which contains three integers u_i, v_i ($1 \le u_i, v_i \le n$) and w_i ($1 \le w_i \le 10^6$), indicating that the i-th edge in the graph has a weight of w_i and connects the u_i-th and the v_i-th vertices.

Note that a simple graph contains no self-loops and no multiple edges.

Example

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

Note

For your convenience, you can copy the `C++` code, which corresponds to the given pseudo code, from the contest website. Save the code as `spfa.cpp`, use `g++ spfa.cpp -O2 -o spfa` to compile it and you will get an executable file named `spfa`. Run `spfa`, feed your output to its standard input and it will print out the **final** value of `cnt`. Given the sample output it will print out 4, which means the sample output is not sufficient to pass the secret test case.

Note that the given code does not check the validity of your output (for example it does not check if your output is really a simple graph). You might still fail the test if your output is invalid, even if the executable prints out a large value.

Problem E. Pass the Ball!

There are n children playing with n balls. Both children and balls are numbered from 1 to n.

Before the game, n integers $p_1, p_2, \cdots, p_n$ are given. In each round of the game, child i will pass the ball he possesses to child p_i. It is guaranteed that no child will pass his ball to himself, which means $p_i \neq i$. Moreover, we also know that after each round, each child will hold exactly one ball.

Let b_i be the ball possessed by child i. At the beginning of the game, child i ($1 \leq i \leq n$) will be carrying ball i, which means $b_i = i$ initially. You're asked to process q queries. For each query you're given an integer k and you need to compute the value of $\sum_{i=1}^{n} i \times b_i$ after k rounds.

Input

There is only one test case for each test file.

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

The second line contains n integers $p_1, p_2, \cdots, p_n$ ($1 \leq p_i \leq n$) indicating how the children pass the balls around.

For the following q lines, the i-th line contains one integer k_i ($1 \leq k_i \leq 10^9$) indicating a query asking for the result after k_i rounds.

Output

For each query output one line containing one integer indicating the answer.

Example

standard input	standard output
4 4 2 4 1 3 1 2 3 4	25 20 25 30

Note

The sample test case is explained below.

Round	b_1	b_2	b_3	b_4	**Answer**
1	3	1	4	2	25
2	4	3	2	1	20
3	2	4	1	3	25
4	1	2	3	4	30

Problem F. Sandpile on Clique

The *Abelian Sandpile Model* is a famous dynamical system displaying self-organized criticality. It has been studied for decades since it was introduced by Per Bak, Chao Tang and Kurt Wiesenfeld in a 1987 paper. The sandpile prediction is of wide interest in physics, computer science, and mathematics, both for its beautiful algebraic structure and for its relevance to applications like load balancing and derandomization of models like internal diffusion-limited aggregation. The sandpile model is related to many other models and physical phenomena, like the rotor-routing model, avalanche models.

In the sandpile model, we are given an undirected graph G whose vertices are indexed from 1 to n. We're also given n integers $a_1, a_2, \cdots, a_n$ where a_i indicates that there are a_i chips placed on vertex i initially. Each turn we will pick an arbitrary vertex v such that the number of chips on v is not smaller than the number of edges connecting v, denoted as d_v. For each neighbor of v, it will receive one chip from v. Therefore, v will lost d_v chips. This process is called firing or toppling. Firing will keep happening until no vertex v has at least d_v chips.

It can be proven that the order of firing doesn't affect the result. Meanwhile, it is also possible that the firing will never terminate. This instance is described as "recurrent". Now you are given a clique and the initial number of chips. Determine whether this instance is a recurrent one. If not, please output the final number of chips for each node respectively.

A clique (also called a complete graph) is a graph where every two vertices are connected with an edge.

Input

There is only one test case in each test file.

The first line of the input contains an integer n ($2 \le n \le 5 \times 10^5$) indicating the size of the clique.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($0 \le a_i \le 10^9$) where a_i indicates the initial number of chips placed on vertex i.

Output

Output one line. If the given sandpile instance will terminate, output n integers separated by a space where the i-th integer indicates the final number of chips on the i-th vertex. Otherwise output "Recurrent" (without quotes) instead.

Please, DO NOT output extra spaces at the end of each line or your solution may be considered incorrect!

Examples

standard input	standard output
5 5 0 3 0 3	3 3 1 3 1
2 1 0	Recurrent

Note

For the first sample test case:

- We can only select vertex 1 at the beginning. The number of chips becomes $\{1, 1, 4, 1, 4\}$.
- We can now select vertex 3 or 5 because both of them have at least 4 chips. We select vertex 3 and the number of chips becomes $\{2, 2, 0, 2, 5\}$. Selecting vertex 5 will lead to the same result.
- We now select vertex 5. The number of chips becomes $\{3, 3, 1, 3, 1\}$. There is no vertex with at least 4 chips so the firing terminates.

For the second sample test case, we can select vertex 1 and 2 repeatedly. The firing never terminates.

Problem G. Cyclic Buffer

There is a cyclic buffer of size n with readers from the 1-st position to the k-th position (both inclusive). Let a_i $(1 \le i \le n)$ be the integer at the i-th position of the buffer initially. What's more, $a_1, a_2, \cdots, a_n$ form a permutation of n.

We're going to visit all the integers from 1 to n (both inclusive) in increasing order. An integer can be visited only when it is residing in positions with readers (that is to say, when it is in the first k positions). In case that an integer cannot be visited, we can shift the whole buffer in either directions any number of times.

- If we shift the buffer to the left once, integers in the i-th position will be moved to the $(i-1)$-th position if $i > 1$, and integer in the 1-st position will be moved to the n-th position.
- If we shift the buffer to the right once, integers in the i-th position will be moved to the $(i+1)$-th position if $i < n$, and integer in the n-th position will be moved to the 1-st position.

What's the minimum number of times to shift the buffer so that we can visit all the integers in increasing order?

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and k $(1 \le k \le n \le 10^6)$ indicating the size of the buffer and the number of readers.

The second line contains n integers $a_1, a_2, \cdots, a_n$ $(1 \le a_i \le n)$ where a_i indicates the integer in the i-th position of the buffer initially.

It's guaranteed that the given n integers form a permutation of n. It's also guaranteed that the sum of n of all test cases will not exceed 10^6.

Output

For each test case output one line containing one integer indicating the minimum number of times to shift the buffer so that all integers can be visited in increasing order.

Example

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

Note

For the first sample test case:

- Shift to the right once so the buffer becomes $\{1, 2, 4, 3, 5\}$. 1 and 2 can now be visited in order as they are in the first 3 positions.
- Shift to the left twice so the buffer becomes $\{4, 3, 5, 1, 2\}$. 3, 4 and 5 can now be visited in order as they are in the first 3 positions.

Problem H. Permutation on Tree

Given a tree with n vertices where vertex r is the root, we say a permutation $p_1, p_2, \cdots, p_n$ of n is good if it satisfies the following constraint:

> Let a_x be the index of x in the permutation (That is, $p_{a_x} = x$). For all $1 \le u, v \le n$, if vertex u is an ancestor of vertex v in the tree, then $a_u < a_v$.

Define the score of a permutation to be $\sum_{i=1}^{n-1} |p_i - p_{i+1}|$ where $|x|$ is the absolute value of x. Calculate the sum of scores of all different good permutations.

Input

There is only one test case in each test file.

The first line contains two integers n and r ($2 \le n \le 200$, $1 \le r \le n$) indicating the size of the tree and the root.

For the following $(n-1)$ lines, the i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n$) indicating an edge connecting vertex u_i and v_i in the tree.

Output

For each test case output one line containing one integer indicating the sum of scores of all different good permutations. As the answer may be large, output the answer modulo $(10^9 + 7)$.

Examples

standard input	standard output
4 2 1 2 2 3 1 4	15
3 1 1 2 2 3	2

Note

For the first sample test case, there are three good permutations: $\{2, 1, 3, 4\}$, $\{2, 1, 4, 3\}$ and $\{2, 3, 1, 4\}$. Their scores are 4, 5 and 6 respectively so the answer is $4 + 5 + 6 = 15$.

For the second sample test case, there is only one good permutation: $\{1, 2, 3\}$. It's score is 2.

Problem I. LCS Spanning Tree

Given a complete undirected graph of n vertices and n strings $s_1, s_2, \cdots, s_n$, the weight of edge connecting vertices i and j is equal to the length of the longest common substring (LCS) between s_i and s_j. Compute the maximum total weight of any spanning tree on this graph.

A substring of a string can be obtained by removing some (possibly zero) characters from the beginning and/or the end of that string. For example, "maca", "aca" and "cau" are all substrings of "macau", while "acu" is not.

Input

There is only one test case in each test file.

The first line of the input contains one integer n ($1 \leq n \leq 2 \times 10^6$) indicating the number of vertices and strings.

For the following n lines, the i-th line contains one string s_i ($1 \leq |s_i| \leq 2 \times 10^6$) consisting only of lowercase English letters.

It's guaranteed that the sum of lengths of all strings will not exceed 2×10^6.

Output

Output one line containing one integer indicating the answer.

Examples

standard input	standard output
4 icpc macau regional contest	4
3 ababa babab aba	7

Problem J. Colorful Tree

Your task is to maintain a colorful tree and process queries.

At the beginning, there is only one vertex numbered 1 with color C on the tree. Then there are q operations of two types coming in order:

- 0 x c d: Add a new vertex indexed $(n+1)$ with color c to the tree, where n is the current number of existing vertices. An edge connecting vertex x and $(n+1)$ with length d will also be added to the tree.
- 1 x c: Change the color of vertex x to c.

After each operation, you should find a pair of vertices u and v $(1 \leq u, v \leq n)$ with **different** colors in the current tree so that the distance between u and v is as large as possible.

The distance between two vertices u and v is the length of the shortest path from u to v on the tree.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line of the input contains two integers q and C $(1 \leq q \leq 5 \times 10^5,\ 1 \leq C \leq q)$ indicating the number of operations and the initial color of vertex 1.

For the following q lines, each line describes an operation taking place in order with 3 or 4 integers.

- If the i-th line contains 4 integers 0, x_i, c_i and d_i $(1 \leq x_i \leq n,\ 1 \leq c_i \leq q,\ 1 \leq d \leq 10^9)$, the i-th operation will add a new vertex $(n+1)$ with color c_i to the tree and connect it to vertex x_i with an edge of length d_i.
- If the i-th line contains 3 integers 1, x_i and c_i $(1 \leq x_i \leq n,\ 1 \leq c_i \leq q)$, the i-th operation will change the color of vertex x_i to c_i.

It's guaranteed that the sum of q of all test cases will not exceed 5×10^5.

Output

For each operation output the maximum distance between two vertices with different colors. If no valid pair exists output 0 instead.

Example

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

Problem K. Link-Cut Tree

BaoBao just learned how to use a data structure called link-cut tree to find cycles in a graph and decided to give it a try. BaoBao is given an undirected graph with n vertices and m edges, where the length of the i-th edge equals 2^i. She needs to find a simple cycle with the smallest length.

A simple cycle is a subgraph of the original graph containing k ($3 \le k \le n$) vertices $a_1, a_2, \cdots, a_k$ and k edges such that for all $1 \le i \le k$ there is an edge connecting vertices a_i and $a_{(i \mod k)+1}$ in the subgraph. The length of a simple cycle is the total length of the edges in the cycle.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

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

For the following m lines, the i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n$) indicating an edge connecting vertices u_i and v_i with length 2^i. There are no self loops nor multiple edges. Note that the graph is not necessarily connected.

It's guaranteed that neither the sum of n nor the sum of m of all test cases will exceed 10^6.

Output

For each test case output one line. If there are no simple cycles in the graph output "-1" (without quotes); Otherwise output k integers separated by a space in increasing order indicating the indices of the edges in the simple cycle with the smallest length. It can be shown that there is at most one answer.

Please, DO NOT output extra spaces at the end of each line, or your solution may be considered incorrect!

Example

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

Note

The first sample test case is shown below. The integers beside the edges are their indices (outside the parentheses) and lengths (inside the parentheses). The simple cycle with the smallest length consists of edges 2, 4, 5 and 6 with a length of $2^2 + 2^4 + 2^5 + 2^6 = 120$.