A. Vasya and Petya's Game

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya and Petya are playing a simple game. Vasya thought of number x between 1 and n, and Petya tries to guess the number.

Petya can ask questions like: "Is the unknown number divisible by number y?".

The game is played by the following rules: first Petya asks **all** the questions that interest him (also, he can ask no questions), and then Vasya responds to each question with a 'yes' or a 'no'. After receiving all the answers Petya should determine the number that Vasya thought of.

Unfortunately, Petya is not familiar with the number theory. Help him find the minimum number of questions he should ask to make a guaranteed guess of Vasya's number, and the numbers y_i, he should ask the questions about.

Input

A single line contains number n ($1 \le n \le 10^3$).

Output

Print the length of the sequence of questions k ($0 \le k \le n$), followed by k numbers — the questions y_i ($1 \le y_i \le n$).

If there are several correct sequences of questions of the minimum length, you are allowed to print any of them.

Examples

input

```
4
```

output

```
3
2 4 3
```

input

```
6
```

output

```
4
2 4 3 5
```

Note

The sequence from the answer to the first sample test is actually correct.

If the unknown number is not divisible by one of the sequence numbers, it is equal to 1.

If the unknown number is divisible by 4, it is 4.

If the unknown number is divisible by 3, then the unknown number is 3.

Otherwise, it is equal to 2. Therefore, the sequence of questions allows you to guess the unknown number. It can be shown that there is no correct sequence of questions of length 2 or shorter.

B. Invariance of Tree

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A tree of size n is an undirected connected graph consisting of n vertices without cycles.

Consider some tree with n vertices. We call a tree *invariant* relative to permutation $p = p_1p_2... p_n$, if for any two vertices of the tree u and v the condition holds: "vertices u and v are connected by an edge if and only if vertices p_u and p_v are connected by an edge".

You are given permutation p of size n. Find some tree size n, invariant relative to the given permutation.

Input

The first line contains number n ($1 \le n \le 10^5$) — the size of the permutation (also equal to the size of the sought tree).

The second line contains permutation p_i ($1 \le p_i \le n$).

Output

If the sought tree does not exist, print "NO" (without the quotes).

Otherwise, print "YES", and then print $n - 1$ lines, each of which contains two integers — the numbers of vertices connected by an edge of the tree you found. The vertices are numbered from 1, the order of the edges and the order of the vertices within the edges does not matter.

If there are multiple solutions, output any of them.

Examples

input

```
4
4 3 2 1
```

output

```
YES
4 1
4 2
1 3
```

input

```
3
3 1 2
```

output

```
NO
```

Note

In the first sample test a permutation transforms edge (4, 1) into edge (1, 4), edge (4, 2) into edge (1, 3) and edge (1, 3) into edge (4, 2). These edges all appear in the resulting tree.

It can be shown that in the second sample test no tree satisfies the given condition.

C. Points on Plane

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

On a plane are n points (x_i, y_i) with integer coordinates between 0 and 10^6. The distance between the two points with numbers a and b is said to be the following value: $\mathrm{dist}(a, b) = |x_a - x_b| + |y_a - y_b|$ (the distance calculated by such formula is called *Manhattan distance*).

We call a hamiltonian path to be some permutation p_i of numbers from 1 to n. We say that the length of this path is value $\sum_{i=1}^{n-1} \mathrm{dist}(p_i, p_{i+1})$.

Find some hamiltonian path with a length of no more than 25×10^8. Note that you do not have to minimize the path length.

Input

The first line contains integer n ($1 \le n \le 10^6$).

The $i + 1$-th line contains the coordinates of the i-th point: x_i and y_i ($0 \le x_i, y_i \le 10^6$).

It is guaranteed that no two points coincide.

Output

Print the permutation of numbers p_i from 1 to n — the sought Hamiltonian path. The permutation must meet the inequality $\sum_{i=1}^{n-1} \mathrm{dist}(p_i, p_{i+1}) \le 25 \times 10^8$.

If there are multiple possible answers, print any of them.

It is guaranteed that the answer exists.

Examples

input

```
5
0 7
8 10
3 4
5 0
9 12
```

output

```
4 3 1 2 5
```

Note

In the sample test the total distance is:

$\mathrm{dist}(4, 3) + \mathrm{dist}(3, 1) + \mathrm{dist}(1, 2) + \mathrm{dist}(2, 5) =$

$(|5 - 3| + |0 - 4|) + (|3 - 0| + |4 - 7|) + (|0 - 8| + |7 - 10|) + (|8 - 9| + |10 - 12|) = 2 + 4 + 3 + 3 + 8 + 3 + 1 + 2 = 26$

D. Flights for Regular Customers

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In the country there are exactly n cities numbered with positive integers from 1 to n. In each city there is an airport is located.

Also, there is the only one airline, which makes m flights. Unfortunately, to use them, you need to be a regular customer of this company, namely, you have the opportunity to enjoy flight i from city a_i to city b_i only if you have already made at least d_i flights before that.

Please note that flight i flies exactly from city a_i to city b_i. It can not be used to fly from city b_i to city a_i. An interesting fact is that there may possibly be recreational flights with a beautiful view of the sky, which begin and end in the same city.

You need to get from city 1 to city n. Unfortunately, you've never traveled by plane before. What minimum number of flights you have to perform in order to get to city n?

Note that the same flight can be used multiple times.

Input

The first line contains two integers, n and m ($2 \le n \le 150$, $1 \le m \le 150$) — the number of cities in the country and the number of flights the company provides.

Next m lines contain numbers a_i, b_i, d_i ($1 \le a_i, b_i \le n$, $0 \le d_i \le 10^9$), representing flight number i from city a_i to city b_i, accessible to only the clients who have made at least d_i flights.

Output

Print "`Impossible`" (without the quotes), if it is impossible to get from city 1 to city n using the airways.

But if there is at least one way, print a single integer — the minimum number of flights you need to make to get to the destination point.

Examples

input

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

output

```
2
```

input

```
2 1
1 2 100500
```

output

```
Impossible
```

input

```
3 3
2 1 0
2 3 6
1 2 0
```

output

```
8
```

E. Painting Edges

time limit per test: 6 seconds
memory limit per test: **600 megabytes**
input: standard input
output: standard output

Note the unusual memory limit for this problem.

You are given an undirected graph consisting of n vertices and m edges. The vertices are numbered with integers from 1 to n, the edges are numbered with integers from 1 to m. Each edge can be unpainted or be painted in one of the k colors, which are numbered with integers from 1 to k. Initially, none of the edges is painted in any of the colors.

You get queries of the form "Repaint edge e_i to color c_i". At any time the graph formed by the edges of the same color must be bipartite. If after the repaint this condition is violated, then the query is considered to be invalid and edge e_i keeps its color. Otherwise, edge e_i is repainted in color c_i, and the query is considered to valid.

Recall that the graph is called *bipartite* if the set of its vertices can be divided into two parts so that no edge connected vertices of the same parts.

For example, suppose you are given a triangle graph, that is a graph with three vertices and edges $(1, 2)$, $(2, 3)$ and $(3, 1)$. Suppose that the first two edges are painted color 1, and the third one is painted color 2. Then the query of "repaint the third edge in color 1" will be incorrect because after its execution the graph formed by the edges of color 1 will not be bipartite. On the other hand, it is possible to repaint the second edge in color 2.

You receive q queries. For each query, you should either apply it, and report that the query is valid, or report that the query is invalid.

Input

The first line contains integers n, m, k, q ($2 \le n \le 5 \cdot 10^5$, $1 \le m, q \le 5 \cdot 10^5$, $1 \le k \le 50$) — the number of vertices, the number of edges, the number of colors and the number of queries.

Then follow m edges of the graph in the form a_i, b_i ($1 \le a_i, b_i \le n$).

Then follow q queries of the form e_i, c_i ($1 \le e_i \le m$, $1 \le c_i \le k$).

It is guaranteed that the graph doesn't contain multiple edges and loops.

Output

For each query print "YES" (without the quotes), if it is valid, or "NO" (without the quotes), if this query destroys the bipartivity of the graph formed by the edges of some color.

Examples

input

```
3 3 2 5
1 2
2 3
1 3
1 1
2 1
3 2
3 1
2 2
```

output

```
YES
YES
YES
NO
YES
```