Codeforces Round #124 (Div. 1)

A. Lexicographically Maximum Subsequence

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

You've got string s, consisting of only lowercase English letters. Find its lexicographically maximum subsequence.

We'll call a non-empty string $s[p_1p_2\ldots p_k] = s_{p_1}s_{p_2}\ldots s_{p_k}(1 \le p_1 < p_2 < \ldots < p_k \le |s|)$ a *subsequence* of string $s = s_1s_2\ldots s_{|s|}$.

String $x = x_1x_2\ldots x_{|x|}$ is *lexicographically larger* than string $y = y_1y_2\ldots y_{|y|}$, if either $|x| > |y|$ and $x_1 = y_1, x_2 = y_2, \ldots, x_{|y|} = y_{|y|}$, or exists such number $r\ (r < |x|, r < |y|)$, that $x_1 = y_1, x_2 = y_2, \ldots, x_r = y_r$ and $x_{r+1} > y_{r+1}$. Characters in lines are compared like their ASCII codes.

Input

The single line contains a non-empty string s, consisting only of lowercase English letters. The string's length doesn't exceed 10^5.

Output

Print the lexicographically maximum subsequence of string s.

Examples

input
ababba
output
bbba

input
abbcbccacbbcbaaba
output
cccccbba

Note

Let's look at samples and see what the sought subsequences look like (they are marked with uppercase bold letters).

The first sample: a**B**a**BBA**

The second sample: abb**C**b**CC**a**C**bb**CB**aa**BA**

B. Infinite Maze

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

We've got a rectangular $n \times m$-cell maze. Each cell is either passable, or is a wall (impassable). A little boy found the maze and cyclically tiled a plane with it so that the plane became an infinite maze. Now on this plane cell (x, y) is a wall if and only if cell $(x \bmod n, y \bmod m)$ is a wall.

In this problem $a \bmod b$ is a remainder of dividing number a by number b.

The little boy stood at some cell on the plane and he wondered whether he can walk infinitely far away from his starting position. From cell (x, y) he can go to one of the following cells: $(x, y - 1)$, $(x, y + 1)$, $(x - 1, y)$ and $(x + 1, y)$, provided that the cell he goes to is not a wall.

Input

The first line contains two space-separated integers n and m ($1 \le n, m \le 1500$) — the height and the width of the maze that the boy used to cyclically tile the plane.

Each of the next n lines contains m characters — the description of the labyrinth. Each character is either a "#", that marks a wall, a ".", that marks a passable cell, or an "S", that marks the little boy's starting point.

The starting point is a passable cell. It is guaranteed that character "S" occurs exactly once in the input.

Output

Print "Yes" (without the quotes), if the little boy can walk infinitely far from the starting point. Otherwise, print "No" (without the quotes).

Examples

input
5 4 ##.# ##S# #..# #.## #..#
output
Yes

input
5 4 ##.# ##S# #..# ..#. #.##
output
No

Note

In the first sample the little boy can go up for infinitely long as there is a "clear path" that goes vertically. He just needs to repeat the following steps infinitely: up, up, left, up, up, right, up.

In the second sample the vertical path is blocked. The path to the left doesn't work, too — the next "copy" of the maze traps the boy.

C. Paint Tree

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

You are given a tree with n vertexes and n points on a plane, no three points lie on one straight line.

Your task is to paint the given tree on a plane, using the given points as vertexes.

That is, you should correspond each vertex of the tree to exactly one point and each point should correspond to a vertex. If two vertexes of the tree are connected by an edge, then the corresponding points should have a segment painted between them. The segments that correspond to non-adjacent edges, should not have common points. The segments that correspond to adjacent edges should have exactly one common point.

Input

The first line contains an integer n ($1 \le n \le 1500$) — the number of vertexes on a tree (as well as the number of chosen points on the plane).

Each of the next $n - 1$ lines contains two space-separated integers u_i and v_i ($1 \le u_i, v_i \le n, u_i \ne v_i$) — the numbers of tree vertexes connected by the i-th edge.

Each of the next n lines contain two space-separated integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$) — the coordinates of the i-th point on the plane. No three points lie on one straight line.

It is guaranteed that under given constraints problem has a solution.

Output

Print n distinct space-separated integers from 1 to n: the i-th number must equal the number of the vertex to place at the i-th point (the points are numbered in the order, in which they are listed in the input).

If there are several solutions, print any of them.

Examples

input
3 1 3 2 3 0 0 1 1 2 0
output
1 3 2

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

Note

The possible solutions for the sample are given below.

D. The Next Good String

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

In problems on strings one often has to find a string with some particular properties. The problem authors were reluctant to waste time on thinking of a name for some string so they called it *good*. A string is good if it doesn't have palindrome substrings longer than or equal to d.

You are given string s, consisting only of lowercase English letters. Find a good string t with length $|s|$, consisting of lowercase English letters, which is lexicographically larger than s. Of all such strings string t must be lexicographically minimum.

We will call a non-empty string $s[a \ldots b] = s_a s_{a+1} \ldots s_b$ $(1 \le a \le b \le |s|)$ a *substring* of string $s = s_1 s_2 \ldots s_{|s|}$.

A non-empty string $s = s_1 s_2 \ldots s_n$ is called *a palindrome* if for all i from 1 to n the following fulfills: $s_i = s_{n-i+1}$. In other words, palindrome read the same in both directions.

String $x = x_1 x_2 \ldots x_{|x|}$ is *lexicographically larger* than string $y = y_1 y_2 \ldots y_{|y|}$, if either $|x| > |y|$ and $x_1 = y_1, x_2 = y_2, \ldots, x_{|y|} = y_{|y|}$, or there exists such number r $(r < |x|, r < |y|)$, that $x_1 = y_1, x_2 = y_2, \ldots, x_r = y_r$ and $x_{r+1} > y_{r+1}$. Characters in such strings are compared like their ASCII codes.

Input

The first line contains integer d $(1 \le d \le |s|)$.

The second line contains a non-empty string s, its length is no more than $4 \cdot 10^5$ characters. The string consists of lowercase English letters.

Output

Print the good string that lexicographically follows s, has the same length and consists of only lowercase English letters. If such string does not exist, print "`Impossible`" (without the quotes).

Examples

input
3 aaaaaaa
output
aabbcaa

input
3 zzyzzzz
output
Impossible

input
4 abbabbbabbb
output
abbbcaaabab

E. Opening Portals

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

Pavel plays a famous computer game. A player is responsible for a whole country and he can travel there freely, complete quests and earn experience.

This country has n cities connected by m bidirectional roads of different lengths so that it is possible to get from any city to any other one. There are portals in k of these cities. At the beginning of the game all portals are closed. When a player visits a portal city, the portal opens. Strange as it is, one can teleport from an open portal to an open one. The teleportation takes no time and that enables the player to travel quickly between rather remote regions of the country.

At the beginning of the game Pavel is in city number 1. He wants to open all portals as quickly as possible. How much time will he need for that?

Input

The first line contains two space-separated integers n and m ($1 \le n \le 10^5$, $0 \le m \le 10^5$) that show how many cities and roads are in the game.

Each of the next m lines contains the description of a road as three space-separated integers x_i, y_i, w_i ($1 \le x_i, y_i \le n$, $x_i \ne y_i$, $1 \le w_i \le 10^9$) — the numbers of the cities connected by the i-th road and the time needed to go from one city to the other one by this road. Any two cities are connected by no more than one road. It is guaranteed that we can get from any city to any other one, moving along the roads of the country.

The next line contains integer k ($1 \le k \le n$) — the number of portals.

The next line contains k space-separated integers p_1, p_2, ..., p_k — numbers of the cities with installed portals. Each city has no more than one portal.

Output

Print a single number — the minimum time a player needs to open all portals.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Examples

input
3 3 1 2 1 1 3 1 2 3 1 3 1 2 3
output
2

input
4 3 1 2 1 2 3 5 2 4 10 3 2 3 4
output
16

input
4 3 1 2 1000000000 2 3 1000000000 3 4 1000000000 4 1 2 3 4
output
3000000000

Note

In the second sample the player has to come to city 2, open a portal there, then go to city 3, open a portal there, teleport back to

city 2 and finally finish the journey in city 4.
