A. New Year Permutation

2 seconds, 256 megabytes

User ainta has a permutation $p_1, p_2, ..., p_n$. As the New Year is coming, he wants to make his permutation as pretty as possible.

Permutation $a_1, a_2, ..., a_n$ is *prettier* than permutation $b_1, b_2, ..., b_n$, if and only if there exists an integer k ($1 \le k \le n$) where $a_1 = b_1, a_2 = b_2, ..., a_{k-1} = b_{k-1}$ and $a_k < b_k$ all holds.

As known, permutation p is so sensitive that it could be only modified by swapping two distinct elements. But swapping two elements is harder than you think. Given an $n \times n$ binary matrix A, user ainta can swap the values of p_i and p_j ($1 \le i, j \le n$, $i \ne j$) if and only if $A_{i,j} = 1$.

Given the permutation p and the matrix A, user ainta wants to know the prettiest permutation that he can obtain.

Input

The first line contains an integer n ($1 \le n \le 300$) — the size of the permutation p.

The second line contains n space-separated integers $p_1, p_2, ..., p_n$ — the permutation p that user ainta has. Each integer between 1 and n occurs exactly once in the given permutation.

Next n lines describe the matrix A. The i-th line contains n characters '`0`' or '`1`' and describes the i-th row of A. The j-th character of the i-th line $A_{i,j}$ is the element on the intersection of the i-th row and the j-th column of A. It is guaranteed that, for all integers i, j where $1 \le i < j \le n$, $A_{i,j} = A_{j,i}$ holds. Also, for all integers i where $1 \le i \le n$, $A_{i,i} = 0$ holds.

Output

In the first line and only line, print n space-separated integers, describing the prettiest permutation that can be obtained.

input

```
7
5 2 4 3 6 7 1
0001001
0000000
0000010
1000001
0000000
0010000
1001000
```

output

```
1 2 4 3 6 7 5
```

input

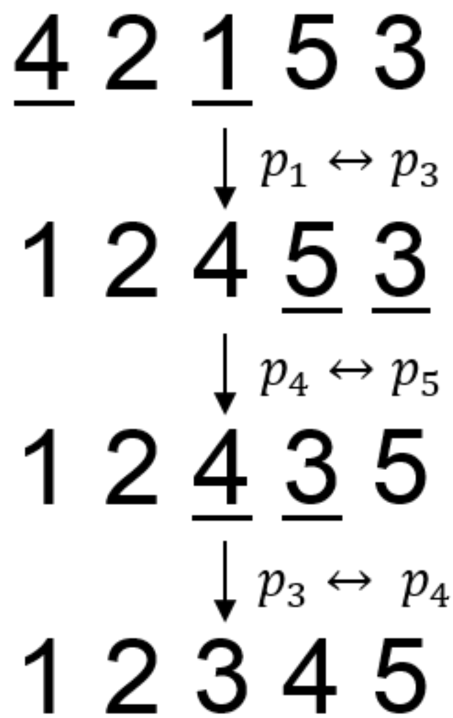
```
5
4 2 1 5 3
00100
00011
10010
01101
01010
```

output

```
1 2 3 4 5
```

In the first sample, the swap needed to obtain the prettiest permutation is: (p_1, p_7).

In the second sample, the swaps needed to obtain the prettiest permutation is $(p_1, p_3), (p_4, p_5), (p_3, p_4)$.



A **permutation** p is a sequence of integers $p_1, p_2, ..., p_n$, consisting of n distinct positive integers, each of them doesn't exceed n. The i-th element of the permutation p is denoted as p_i. The size of the permutation p is denoted as n.

B. Correct Bracket Sequence Editor

2 seconds, 256 megabytes

Recently Polycarp started to develop a text editor that works only with correct bracket sequences (abbreviated as CBS).

Note that a bracket sequence is correct if it is possible to get a correct mathematical expression by adding "+"-s and "1"-s to it. For example, sequences "(())()", "()" and "(()(()))" are correct, while ")(", "(()" and "(()))(" are not. Each bracket in CBS has a pair. For example, in "(()(()))":

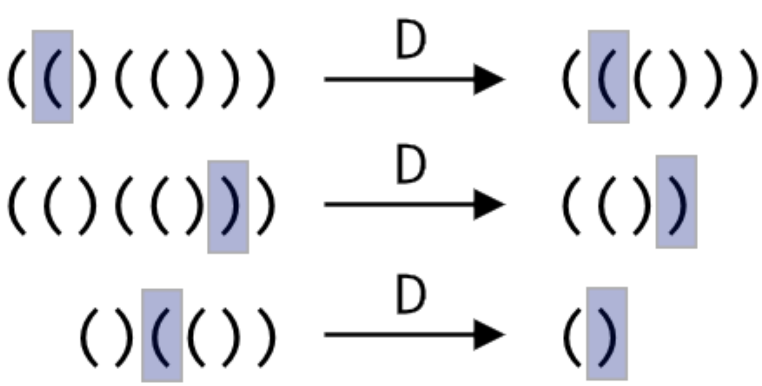
- 1st bracket is paired with 8th,
- 2d bracket is paired with 3d,
- 3d bracket is paired with 2d,
- 4th bracket is paired with 7th,
- 5th bracket is paired with 6th,
- 6th bracket is paired with 5th,
- 7th bracket is paired with 4th,
- 8th bracket is paired with 1st.

Polycarp's editor currently supports only three operations during the use of CBS. The cursor in the editor takes the whole position of one of the brackets (not the position between the brackets!). There are three operations being supported:

- «L» — move the cursor one position to the left,
- «R» — move the cursor one position to the right,
- «D» — delete the bracket in which the cursor is located, delete the bracket it's paired to and all brackets between them (that is, delete a substring between the bracket in which the cursor is located and the one it's paired to).

After the operation "D" the cursor moves to the nearest bracket to the right (of course, among the non-deleted). If there is no such bracket (that is, the suffix of the CBS was deleted), then the cursor moves to the nearest bracket to the left (of course, among the non-deleted).

There are pictures illustrated several usages of operation "D" below.



All incorrect operations (shift cursor over the end of CBS, delete the whole CBS, etc.) are not supported by Polycarp's editor.

Polycarp is very proud of his development, can you implement the functionality of his editor?

Input

The first line contains three positive integers n, m and p ($2 \le n \le 500\,000$, $1 \le m \le 500\,000$, $1 \le p \le n$) — the number of brackets in the correct bracket sequence, the number of operations and the initial position of cursor. Positions in the sequence are numbered from left to right, starting from one. It is guaranteed that n is even.

It is followed by the string of n characters "`(`" and "`)`" forming the correct bracket sequence.

Then follow a string of m characters "`L`", "`R`" and "`D`" — a sequence of the operations. Operations are carried out one by one from the first to the last. It is guaranteed that the given operations never move the cursor outside the bracket sequence, as well as the fact that after all operations a bracket sequence will be non-empty.

Output

Print the correct bracket sequence, obtained as a result of applying all operations to the initial sequence.

input

```
8 4 5
(())()()
RDLD
```

output

```
()
```

input

```
12 5 3
((()())(()))
RRDLD
```

output

```
(()(()))
```

input

```
8 8 8
(())()()
LLLLLLDD
```

output

```
()()
```

In the first sample the cursor is initially at position 5. Consider actions of the editor:

1. command "`R`" — the cursor moves to the position 6 on the right;
2. command "`D`" — the deletion of brackets from the position 5 to the position 6. After that CBS takes the form `(())()`, the cursor is at the position 5;
3. command "`L`" — the cursor moves to the position 4 on the left;
4. command "`D`" — the deletion of brackets from the position 1 to the position 4. After that CBS takes the form `()`, the cursor is at the position 1.

Thus, the answer is equal to `()`.

C. Roads not only in Berland

2 seconds, 256 megabytes

Berland Government decided to improve relations with neighboring countries. First of all, it was decided to build new roads so that from each city of Berland and neighboring countries it became possible to reach all the others. There are n cities in Berland and neighboring countries in total and exactly $n - 1$ two-way roads. Because of the recent financial crisis, the Berland Government is strongly pressed for money, so to build a new road it has to close some of the existing ones. Every day it is possible to close one existing road and immediately build a new one. Your task is to determine how many days would be needed to rebuild roads so that from each city it became possible to reach all the others, and to draw a plan of closure of old roads and building of new ones.

Input

The first line contains integer n ($2 \le n \le 1000$) — amount of cities in Berland and neighboring countries. Next $n - 1$ lines contain the description of roads. Each road is described by two space-separated integers a_i, b_i ($1 \le a_i, b_i \le n, a_i \ne b_i$) — pair of cities, which the road connects. It can't be more than one road between a pair of cities. No road connects the city with itself.

Output

Output the answer, number t — what is the least amount of days needed to rebuild roads so that from each city it became possible to reach all the others. Then output t lines — the plan of closure of old roads and building of new ones. Each line should describe one day in the format `i j u v` — it means that road between cities `i` and `j` became closed and a new road between cities `u` and `v` is built. Cities are numbered from 1. If the answer is not unique, output any.

input
2 1 2
output
0

input
7 1 2 2 3 3 1 4 5 5 6 6 7
output
1 3 1 3 7

D. st-Spanning Tree

4 seconds, 256 megabytes

You are given an undirected connected graph consisting of n vertices and m edges. There are no loops and no multiple edges in the graph.

You are also given two distinct vertices s and t, and two values d_s and d_t. Your task is to build any spanning tree of the given graph (note that the graph is not weighted), such that the degree of the vertex s doesn't exceed d_s, and the degree of the vertex t doesn't exceed d_t, or determine, that there is no such spanning tree.

The *spanning tree* of the graph G is a subgraph which is a tree and contains all vertices of the graph G. In other words, it is a connected graph which contains $n - 1$ edges and can be obtained by removing some of the edges from G.

The degree of a vertex is the number of edges incident to this vertex.

Input

The first line of the input contains two integers n and m ($2 \le n \le 200\,000$, $1 \le m \le min(400\,000, n \cdot (n - 1) / 2)$) — the number of vertices and the number of edges in the graph.

The next m lines contain the descriptions of the graph's edges. Each of the lines contains two integers u and v ($1 \le u, v \le n, u \ne v$) — the ends of the corresponding edge. It is guaranteed that the graph contains no loops and no multiple edges and that it is connected.

The last line contains four integers s, t, d_s, d_t ($1 \le s, t \le n, s \ne t, 1 \le d_s, d_t \le n - 1$).

Output

If the answer doesn't exist print "`No`" (without quotes) in the only line of the output.

Otherwise, in the first line print "`Yes`" (without quotes). In the each of the next $(n - 1)$ lines print two integers — the description of the edges of the spanning tree. Each of the edges of the spanning tree must be printed exactly once.

You can output edges in any order. You can output the ends of each edge in any order.

If there are several solutions, print any of them.

input

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

output

```
Yes
3 2
1 3
```

input

```
7 8
7 4
1 3
5 4
5 7
3 2
2 4
6 1
1 2
6 4 1 4
```

output

```
Yes
1 3
5 7
3 2
7 4
2 4
6 1
```

E. New Year Domino

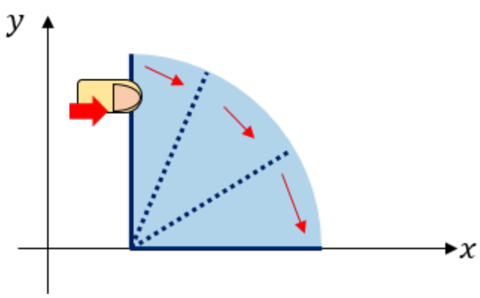
2 seconds, 256 megabytes

Celebrating the new year, many people post videos of falling dominoes; Here's a list of them:
`https://www.youtube.com/results?search_query=New+Years+Dominos`

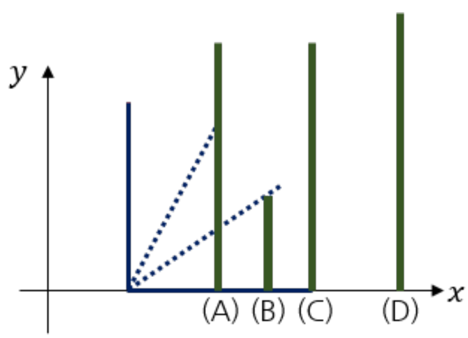
User ainta, who lives in a 2D world, is going to post a video as well.

There are n dominoes on a 2D Cartesian plane. i-th domino ($1 \le i \le n$) can be represented as a line segment which is parallel to the y-axis and whose length is l_i. The lower point of the domino is on the x-axis. Let's denote the x-coordinate of the i-th domino as p_i. Dominoes are placed one after another, so $p_1 < p_2 < ... < p_{n-1} < p_n$ holds.

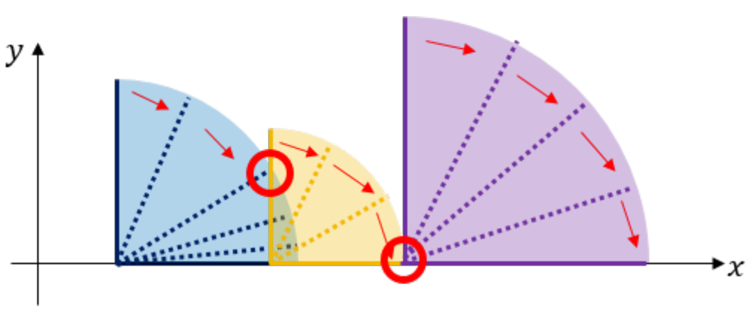
User ainta wants to take a video of falling dominoes. To make dominoes fall, he can push a single domino to the right. Then, the domino will fall down drawing a circle-shaped orbit until the line segment totally overlaps with the x-axis.



Also, if the s-th domino touches the t-th domino while falling down, the t-th domino will also fall down towards the right, following the same procedure above. Domino s touches domino t if and only if the segment representing s and t intersects.



See the picture above. If he pushes the leftmost domino to the right, it falls down, touching dominoes (A), (B) and (C). As a result, dominoes (A), (B), (C) will also fall towards the right. However, domino (D) won't be affected by pushing the leftmost domino, but eventually it will fall because it is touched by domino (C) for the first time.



The picture above is an example of falling dominoes. Each red circle denotes a touch of two dominoes.

User ainta has q plans of posting the video. j-th of them starts with pushing the x_j-th domino, and lasts until the y_j-th domino falls. But sometimes, it could be impossible to achieve such plan, so he has to lengthen some dominoes. It costs one dollar to increase the length of a single domino by 1. User ainta wants to know, for each plan, the minimum cost needed to achieve it. Plans are processed independently, i. e. if domino's length is increased in some plan, it doesn't affect its length in other plans. Set of dominos that will fall except x_j-th domino and y_j-th domino doesn't matter, but the initial push should be on domino x_j.

Input

The first line contains an integer n $(2 \le n \le 2 \times 10^5)$— the number of dominoes.

Next n lines describe the dominoes. The i-th line $(1 \le i \le n)$ contains two space-separated integers p_i, l_i $(1 \le p_i, l_i \le 10^9)$— the x-coordinate and the length of the i-th domino. It is guaranteed that $p_1 < p_2 < \ldots < p_{n-1} < p_n$.

The next line contains an integer q $(1 \le q \le 2 \times 10^5)$ — the number of plans.

Next q lines describe the plans. The j-th line $(1 \le j \le q)$ contains two space-separated integers x_j, y_j $(1 \le x_j < y_j \le n)$. It means the j-th plan is, to push the x_j-th domino, and shoot a video until the y_j-th domino falls.

Output

For each plan, print a line containing the minimum cost needed to achieve it. If no cost is needed, print 0.

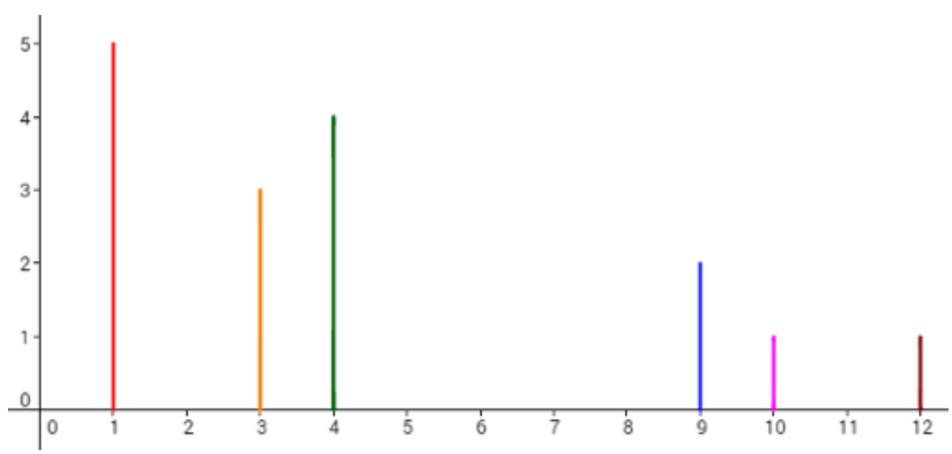
input

```
6
1 5
3 3
4 4
9 2
10 1
12 1
4
1 2
2 4
2 5
2 6
```

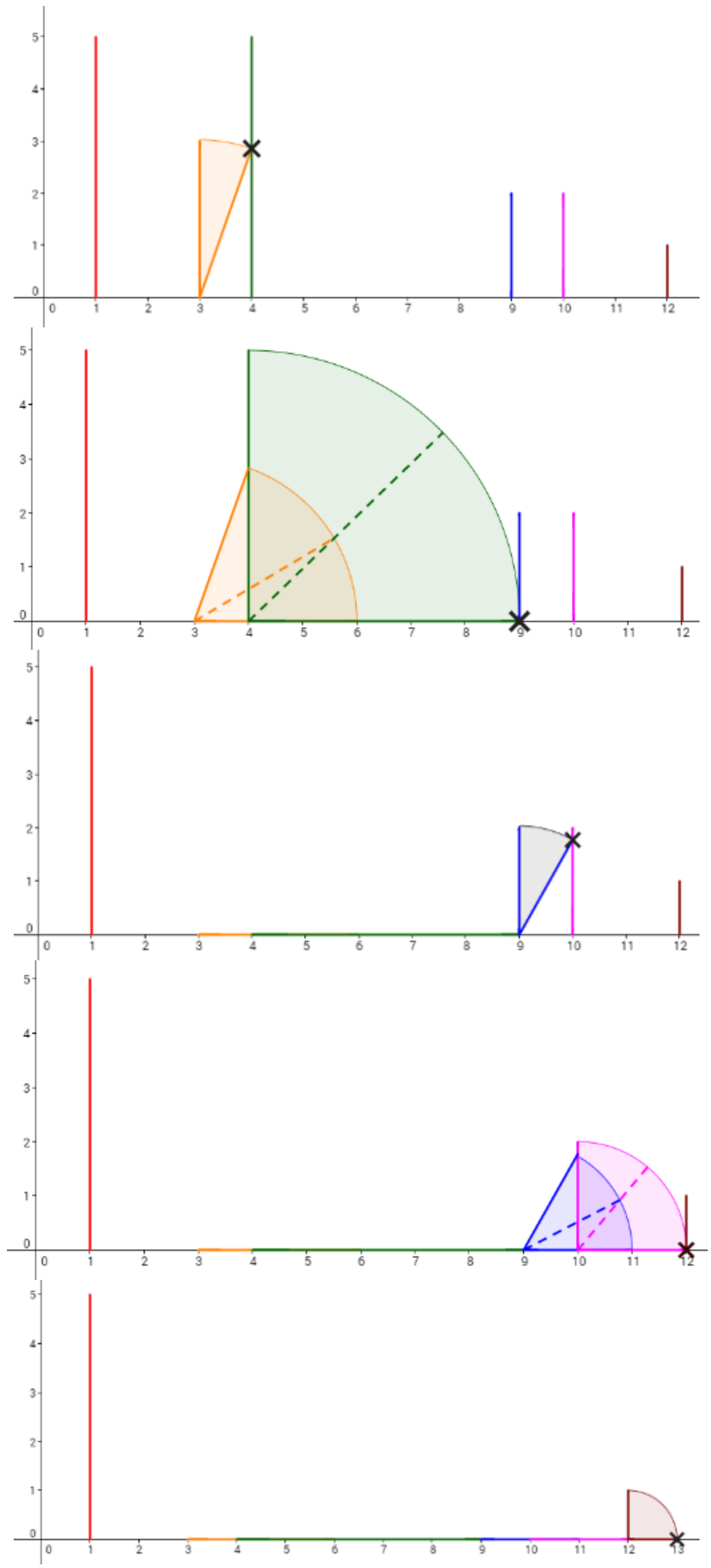
output

```
0
1
1
2
```

Consider the example. The dominoes are set like the picture below.



Let's take a look at the 4th plan. To make the 6th domino fall by pushing the 2nd domino, the length of the 3rd domino (whose x-coordinate is 4) should be increased by 1, and the 5th domino (whose x-coordinate is 9) should be increased by 1 (other option is to increase 4th domino instead of 5th also by 1). Then, the dominoes will fall like in the picture below. Each cross denotes a touch between two dominoes.



F. Imbalance Value of a Tree

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

You are given a tree T consisting of n vertices. A number is written on each vertex; the number written on vertex i is a_i. Let's denote the function $I(x, y)$ as the difference between maximum and minimum value of a_i on a simple path connecting vertices x and y.

Your task is to calculate $\sum_{i=1}^{n} \sum_{j=i}^{n} I(i, j)$.

Input

The first line contains one integer number n ($1 \le n \le 10^6$) — the number of vertices in the tree.

The second line contains n integer numbers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^6$) — the numbers written on the vertices.

Then $n - 1$ lines follow. Each line contains two integers x and y denoting an edge connecting vertex x and vertex y ($1 \le x, y \le n, x \ne y$). It is guaranteed that these edges denote a tree.

Output

Print one number equal to $\sum_{i=1}^{n} \sum_{j=i}^{n} I(i, j)$.

Example

input
4 2 2 3 1 1 2 1 3 1 4

output
6