Problem Set 11: Network Flows

15-295 Spring 2018

A. Soldier and Traveling

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

In the country there are n cities and m bidirectional roads between them. Each city has an army. Army of the i-th city consists of a_i soldiers. Now soldiers roam. After roaming each soldier has to either stay in his city or to go to the one of neighboring cities by at **moving along at most one road**.

Check if is it possible that after roaming there will be exactly b_i soldiers in the i-th city.

Input

First line of input consists of two integers n and m ($1 \le n \le 100$, $0 \le m \le 200$).

Next line contains n integers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 100$).

Next line contains n integers $b_1, b_2, ..., b_n$ ($0 \le b_i \le 100$).

Then m lines follow, each of them consists of two integers p and q ($1 \le p, q \le n, p \ne q$) denoting that there is an undirected road between cities p and q.

It is guaranteed that there is at most one road between each pair of cities.

Output

If the conditions can not be met output single word "NO".

Otherwise output word "YES" and then n lines, each of them consisting of n integers. Number in the i-th line in the j-th column should denote how many soldiers should road from city i to city j (if $i \ne j$) or how many soldiers should stay in city i (if $i = j$).

If there are several possible answers you may output any of them.

Examples

input

Copy

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

output

```
YES
1 0 0 0
2 0 0 0
0 5 1 0
0 0 2 1
```

input

Copy

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

output

```
NO
```

Problem B. Heavy Chain Clusterization

Input file:	`heavy.in`
Output file:	`heavy.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

A group of biologists is trying to find a cure for a viral disease. They have tried many antibodies of various origins that could potentially fight the viral antigens, and have selected n antibodies that seem to work best during experiments.

Each antibody is identified by its *heavy chain* — a sequence of amino acids.

The set of antibodies form a *similarity cluster*, if at least one of the following holds:

- k-prefixes (first k amino acids) of all their heavy chains are equal;
- k-suffixes (last k amino acids) of all their heavy chains are equal.

In order to simplify the future research, biologists want to group antibodies to similarity clusters.

You need to split the given antibodies to a minimum number of similarity clusters.

Input

The first line contains two integers n and k — the number of heavy chains and the length of sequence of amino acids to coincide ($1 \le n \le 5\,000$, $1 \le k \le 550$).

The following n lines contain sequences of amino acids that form heavy chains of antibodies. Each amino acid described with an uppercase English letter. Each heavy chain contains at least k and no more than 550 amino acids.

Output

The first line of output must contain a single integer — the minimum number of similarity clusters. The following lines must contain descriptions of clusters, one per line.

Each description starts with m_i — the number of antibodies in the cluster and is followed by m_i integers — numbers of these antibodies. Antibodies are numbered in the order of appearance in the input starting from one.

Each antibody must be present in exactly one cluster.

Examples

heavy.in	heavy.out

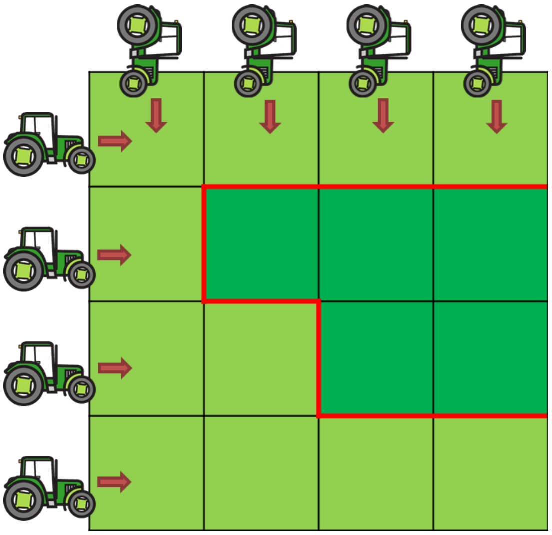
4 1
AA
AB
BB
BA
	2
	2 1 2
	2 3 4

3 2
ABA
BAB
XY
	3
	1 1
	1 2
	1 3

C. Landscaping

Preparations for a good harvest in Spring start now and farmer John is preparing his field for a good season. He went over-budget last year, as the tractors moving up and down the hills needed more fuel than he expected.

When harvesting, his tractors need to move both horizontally and vertically across all the land. In the image, you can see a region at low altitude in light green and a region at high altitude in dark green. When harvesting, his tractors will have to cross all the hills marked in red and they will have to go up or down 8 times.



This year, he is wondering whether he should level some parts of his field before sowing in order to lower his harvesting costs later on. Can you help him decide where the bulldozers should work in order to lower his costs?

Farmer John knows that his tractors need A additional euros when moving between adjacent patches of land at different heights. He can also pay B euros to either increase or decrease the height of any patch in his field.

What is the minimum amount of money he will have to pay this season?

Task

Given a description of the field, the price to change the height of a patch of land and the price his tractors pay when moving between adjacent patches, the goal is to find out the minimum amount that farmer John will have to pay this year.

Input

The first line consists of 4 space separated integers, N, M, A and B. N and M represent the dimensions of his $N \times M$ field, A represents the cost to move between adjacent patches of land at different levels and B is the cost to change any patch of land.

The next N lines each have M characters and represent farmer John’s field. A ’.’ signals a patch of land at a low level and a ’#’ represents a patch of land at a high level.

Constraints

$1 \leq N, M \leq 50$	Size of the field.
$1 \leq A, B \leq 100\,000$	Cost to change any height or to move between adjacent patches.

Output

You should output a single line with a single integer representing the minimum amount of money that farmer John will have to pay.

Sample Input

```
5 4 1000 2000
...#
#..#
...#
##..
###.
```

Sample Output

```
11000
```

Output Explanation

Farmer John has a 5 × 4 field. Moving between adjacent patches at a different level requires €1000 in fuel, while changing the height of a patch costs €2000.

Farmer John needs €11000: €2000 to change the isolated patch to a lower level and €9000 for the fuel needed to move between patches of land at different levels.

Not changing any patch would cost him €12000, changing all the high patches to low would cost him €18000, and changing all the low patches to high would cost him €22000.

D. Binary Tree on Plane

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A root tree is a directed acyclic graph that contains one node (root), from which there is exactly one path to any other node.

A root tree is binary if each node has at most two outgoing arcs.

When a binary tree is painted on the plane, all arcs should be directed from top to bottom. That is, each arc going from u to v must meet the condition $y_u > y_v$.

You've been given the coordinates of all tree nodes. Your task is to connect these nodes by arcs so as to get the binary root tree and make the total length of the arcs minimum. All arcs of the built tree must be directed from top to bottom.

Input

The first line contains a single integer n $(2 \le n \le 400)$ — the number of nodes in the tree. Then follow n lines, two integers per line: x_i, y_i $(|x_i|, |y_i| \le 10^3)$ — coordinates of the nodes. It is guaranteed that all points are distinct.

Output

If it is impossible to build a binary root tree on the given points, print "`-1`". Otherwise, print a single real number — the total length of the arcs in the minimum binary tree. The answer will be considered correct if the absolute or relative error doesn't exceed 10^{-6}.

Examples

input Copy

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

output

```
3.650281539872885
```

input Copy

```
4
0 0
1 0
2 1
2 0
```

output

```
-1
```

E. Anti-Palindromize

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

A string a of length m is called *antipalindromic* iff m is even, and for each i $(1 \le i \le m)$ $a_i \neq a_{m - i + 1}$.

Ivan has a string s consisting of n lowercase Latin letters; n is even. He wants to form some string t that will be an *antipalindromic* permutation of s. Also Ivan has denoted the *beauty* of index i as b_i, and the *beauty* of t as the sum of b_i among all indices i such that $s_i = t_i$.

Help Ivan to determine maximum possible *beauty* of t he can get.

Input

The first line contains one integer n ($2 \le n \le 100$, n is even) — the number of characters in s.

The second line contains the string s itself. It consists of only lowercase Latin letters, and it is guaranteed that its letters can be reordered to form an *antipalindromic* string.

The third line contains n integer numbers b_1, b_2, ..., b_n $(1 \le b_i \le 100)$, where b_i is the *beauty* of index i.

Output

Print one number — the maximum possible *beauty* of t.

Examples

input Copy

```
8
abacabac
1 1 1 1 1 1 1 1
```

output

```
8
```

input Copy

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

output

```
26
```

input Copy

```
8
abacabca
1 2 3 4 4 3 2 1
```

output

```
17
```