Codeforces Round #385 (Div. 2)

A. Hongcow Learns the Cyclic Shift

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

Hongcow is learning to spell! One day, his teacher gives him a word that he needs to learn to spell. Being a dutiful student, he immediately learns how to spell the word.

Hongcow has decided to try to make new words from this one. He starts by taking the word he just learned how to spell, and moves the last character of the word to the beginning of the word. He calls this a *cyclic shift*. He can apply cyclic shift many times. For example, consecutively applying cyclic shift operation to the word "abracadabra" Hongcow will get words "aabracadabr", "raabracadab" and so on.

Hongcow is now wondering how many distinct words he can generate by doing the cyclic shift arbitrarily many times. The initial string is also counted.

Input

The first line of input will be a single string s ($1 \leq |s| \leq 50$), the word Hongcow initially learns how to spell. The string s consists only of lowercase English letters ('a'–'z').

Output

Output a single integer equal to the number of distinct strings that Hongcow can obtain by applying the cyclic shift arbitrarily many times to the given string.

Examples

input
abcd
output
4

input
bbb
output
1

input
yzyz
output
2

Note

For the first sample, the strings Hongcow can generate are "abcd", "dabc", "cdab", and "bcda".

For the second sample, no matter how many times Hongcow does the cyclic shift, Hongcow can only generate "bbb".

For the third sample, the two strings Hongcow can generate are "yzyz" and "zyzy".

B. Hongcow Solves A Puzzle

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

Hongcow likes solving puzzles.

One day, Hongcow finds two identical puzzle pieces, with the instructions "make a rectangle" next to them. The pieces can be described by an n by m grid of characters, where the character 'X' denotes a part of the puzzle and '.' denotes an empty part of the grid. It is guaranteed that the puzzle pieces are one 4-connected piece. See the input format and samples for the exact details on how a jigsaw piece will be specified.

The puzzle pieces are very heavy, so Hongcow **cannot rotate or flip** the puzzle pieces. However, he is allowed to move them in any directions. The puzzle pieces also **cannot overlap**.

You are given as input the description of one of the pieces. Determine if it is possible to make a rectangle from two identical copies of the given input. The rectangle should be solid, i.e. there should be no empty holes inside it or on its border. Keep in mind that Hongcow is not allowed to flip or rotate pieces and they cannot overlap, i.e. no two 'X' from different pieces can share the same position.

Input

The first line of input will contain two integers n and m ($1 \le n, m \le 500$), the dimensions of the puzzle piece.

The next n lines will describe the jigsaw piece. Each line will have length m and will consist of characters '.' and 'X' only. 'X' corresponds to a part of the puzzle piece, '.' is an empty space.

It is guaranteed there is at least one 'X' character in the input and that the 'X' characters form a 4-connected region.

Output

Output "YES" if it is possible for Hongcow to make a rectangle. Output "NO" otherwise.

Examples

input
2 3 XXX XXX
output
YES

input
2 2 .X XX
output
NO

input
5 5 ....X.. ....
output
YES

Note

For the first sample, one example of a rectangle we can form is as follows

```
111222
111222
```

For the second sample, it is impossible to put two of those pieces without rotating or flipping to form a rectangle.

In the third sample, we can shift the first tile by one to the right, and then compose the following rectangle:

```
.....
..XX.
.....
.....
.....
```

C. Hongcow Builds A Nation

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

Hongcow is ruler of the world. As ruler of the world, he wants to make it easier for people to travel by road within their own countries.

The world can be modeled as an undirected graph with n nodes and m edges. k of the nodes are home to the governments of the k countries that make up the world.

There is at most one edge connecting any two nodes and no edge connects a node to itself. Furthermore, for any two nodes corresponding to governments, **there is no path between those two nodes**. Any graph that satisfies all of these conditions is *stable*.

Hongcow wants to add as many edges as possible to the graph while keeping it stable. Determine the maximum number of edges Hongcow can add.

Input

The first line of input will contain three integers n, m and k ($1 \le n \le 1\,000$, $0 \le m \le 100\,000$, $1 \le k \le n$) — the number of vertices and edges in the graph, and the number of vertices that are homes of the government.

The next line of input will contain k integers c_1, c_2, ..., c_k ($1 \le c_i \le n$). These integers will be pairwise distinct and denote the nodes that are home to the governments in this world.

The following m lines of input will contain two integers u_i and v_i ($1 \le u_i$, $v_i \le n$). This denotes an undirected edge between nodes u_i and v_i.

It is guaranteed that the graph described by the input is stable.

Output

Output a single integer, the maximum number of edges Hongcow can add to the graph while keeping it stable.

Examples

input
4 1 2 1 3 1 2
output
2

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

Note

For the first sample test, the graph looks like this:

Vertices 1 and 3 are special. The optimal solution is to connect vertex 4 to vertices 1 and 2. This adds a total of 2 edges. We cannot add any more edges, since vertices 1 and 3 cannot have any path between them.
For the second sample test, the graph looks like this:

We cannot add any more edges to this graph. Note that we are not allowed to add self-loops, and the graph must be simple.

D. Hongcow's Game

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

This is an interactive problem. In the interaction section below you will see the information about flushing the output.

In this problem, you will be playing a game with Hongcow. How lucky of you!

Hongcow has a hidden n by n matrix M. Let $M_{i,j}$ denote the entry i-th row and j-th column of the matrix. The rows and columns are labeled from 1 to n.

The matrix entries are between 0 and 10^9. In addition, $M_{i,i} = 0$ for all valid i. Your task is to find the minimum value along each row, excluding diagonal elements. Formally, for each i, you must find .

To do this, you can ask Hongcow some questions.

A question consists of giving Hongcow a subset of distinct indices $\{w_1, w_2, ..., w_k\}$, with $1 \leq k \leq n$. Hongcow will respond with n integers. The i-th integer will contain the minimum value of $min_{1 \leq j \leq k} M_{i,w_j}$.

You may only ask Hongcow at most 20 questions — he thinks you only need that many questions answered.

When you are ready to answer, print out a single integer -1 on its own line, then n integers on the next line. The i-th integer should be the minimum value in the i-th row of the matrix, excluding the i-th element. Do not forget to flush the final answer as well. Printing the answer does not count as asking a question.

You will get `Wrong Answer` verdict if

- Your question or answers are not in the format described in this statement.
- You ask strictly more than 20 questions.
- Your question contains duplicate indices.
- The value of k in your question does not lie in the range from 1 to n, inclusive.
- Your final answer is not correct.

You will get `Idleness Limit Exceeded` if you don't print anything or if you forget to flush the output, including for the final answer (more info about flushing output below).

Input

The first line of input will contain a single integer n ($2 \leq n \leq 1,000$).

Output

To print the final answer, print out the string `-1` on its own line. Then, the next line should contain n integers. The i-th integer should be the minimum value of the i-th row of the matrix, excluding elements on the diagonal. **Do not forget to flush your answer!**

Interaction

To ask a question, print out a single integer k on its own line, denoting the size of your subset. Then, the next line should contain k integers $w_1, w_2, ... w_k$. Note, you must flush your output to get a response.

Hongcow will respond by printing out a line with n integers. The i-th integer in this line represents the minimum value of M_{i,w_j} where j is between 1 and k.

You may only ask a question at most 20 times, otherwise, you will get `Wrong Answer`.

To flush you can use (just after printing an integer and end-of-line):

- `fflush(stdout)` in C++;
- `System.out.flush()` in Java;
- `stdout.flush()` in Python;
- `flush(output)` in Pascal;
- See the documentation for other languages.

Hacking To hack someone, use the following format

```
n
M_{1,1} M_{1,2} ... M_{1,n}
M_{2,1} M_{2,2} ... M_{2,n}
...
M_{n,1} M_{n,2} ... M_{n,n}
```

Of course, contestant programs will not be able to see this input.

Examples

input

```
3
0 0 0
2 7 0
0 0 4
3 0 8
0 5 4
```

output

```
3
1 2 3
1
3
2
1 2
1
2
1
1
-1
2 5 4
```

input

```
2
0 0
0 0
```

output

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

Note

In the first sample, Hongcow has the hidden matrix

```
[
 [0, 3, 2],
 [5, 0, 7],
 [4, 8 ,0],
]
```

Here is a more readable version demonstrating the interaction. The column on the left represents Hongcow, while the column on the right represents the contestant.

```
3
                3
                1 2 3
0 0 0
                1
                3
2 7 0
                2
                1 2
0 0 4
                1
                2
3 0 8
                1
                1
0 5 4
                -1
                2 5 4
```

For the second sample, it is possible for off-diagonal elements of the matrix to be zero.

E. Hongcow Buys a Deck of Cards

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

One day, Hongcow goes to the store and sees a brand new deck of n special cards. Each individual card is either red or blue. He decides he wants to buy them immediately. To do this, he needs to play a game with the owner of the store.

This game takes some number of turns to complete. On a turn, Hongcow may do one of two things:

- Collect tokens. Hongcow collects 1 red token **and** 1 blue token by choosing this option (thus, 2 tokens in total per one operation).
- Buy a card. Hongcow chooses some card and spends tokens to purchase it as specified below.

The i-th card requires r_i red resources and b_i blue resources. Suppose Hongcow currently has A red cards and B blue cards. Then, the i-th card will require Hongcow to spend $max(r_i - A, 0)$ red tokens, and $max(b_i - B, 0)$ blue tokens. Note, only tokens disappear, but the cards stay with Hongcow forever. Each card can be bought only once.

Given a description of the cards and their costs determine the minimum number of turns Hongcow needs to purchase all cards.

Input

The first line of input will contain a single integer n ($1 \le n \le 16$).

The next n lines of input will contain three tokens c_i, r_i and b_i. c_i will be 'R' or 'B', denoting the color of the card as red or blue. r_i will be an integer denoting the amount of red resources required to obtain the card, and b_i will be an integer denoting the amount of blue resources required to obtain the card ($0 \le r_i, b_i \le 10^7$).

Output

Output a single integer, denoting the minimum number of turns needed to acquire all the cards.

Examples

input
3 R 0 1 B 1 0 R 1 1
output
4

input
3 R 3 0 R 2 0 R 1 0
output
6

Note

For the first sample, Hongcow's four moves are as follows:

1. Collect tokens
2. Buy card 1
3. Buy card 2
4. Buy card 3

Note, at the fourth step, Hongcow is able to buy card 3 because Hongcow already has one red and one blue card, so we don't need to collect tokens.

For the second sample, one optimal strategy is as follows:

1. Collect tokens
2. Collect tokens
3. Buy card 2
4. Collect tokens
5. Buy card 3
6. Buy card 1

At the fifth step, even though Hongcow has a red token, Hongcow doesn't actually need to spend it, since Hongcow has a red card already.