COCI 2018/2019

Round #1, October 20th, 2018

Tasks

Task	Time limit	Memory limit	Score
Nadan	1 s	64 MB	50
Zamjena	1 s	64 MB	70
Cipele	1 s	256 MB	90
Strah	1 s	256 MB	110
Teoretičar	6 s	256 MB	130
Total			450

Alongside being a palindrome, Nadan is also a successful businessman who finances young software developers when they start working on their projects. This year, he decided to distribute *K* kunas (Croatian currency) to *N* projects in a way that each project gets at least one kuna and all projects get a different amounts of kunas. This will always be possible.

Write a program which will, for a given *N* and *K*, find one possible distribution of *K* kunas to *N* projects.

INPUT

The first line contains a positive integer *K* ($100 \le K \le 1\ 000\ 000$), number from the task description.
The second line contains a positive integer *N* ($1 \le N \le 100$), number from the task description.

OUTPUT

For a chosen money distribution, output the amount of money the first project will get in the first line, the amount of money the second project will get in the second line and so on until the *N*-th line where you should output the amount of money the *N*-th project will get.

SCORING

In test cases worth 25 points, it will hold that *N* = 5.
If your solution outputs a distribution where all projects didn’t get a different amount of kunas, you will get 3 out of the possible 5 points for that test case.

SAMPLE TESTS

input

```
100
5
```

output

```
10
30
20
5
35
```

input

```
200
1
```

output

```
200
```

input

```
100
4
```

output

```
21
36
12
31
```

Vlatko likes to play with integer arrays. He wrote two arrays of *N* elements on a piece of paper, each element being either a positive integer or a sequence of lowercase letters of the English alphabet representing a variable. A variable can be replaced with an arbitrary integer. It's possible that both arrays contain the same variable or the same variable occurs multiple times in an array. If that’s the case, each occurence of the variable has to be replaced with the same integer in both arrays.

Vlatko wonders if it's possible to replace all variables with some integer values in such a way that the two arrays become equal. Two arrays are considered equal if the numbers on the same positions in the arrays are equal.

INPUT

The first line contains a positive integer *N* (1 ≤ *N* ≤ 50 000), the number of elements in each array.
The second line contains *N* elements of the first array.
The third line contains *N* elements of the second array.
Each element in both arrays can either be:

- a positive integer less than 1 000 or
- a sequence of lowercase letters of the English alphabet (no longer than 10 characters) which represents a variable.

OUTPUT

If it's possible to replace all variables with integer values in a way that the two arrays become equal, print “DA” (Croatian for yes, without the quotation marks). Otherwise print “NE” (Croatian for no).

SCORING

In test cases worth 20% of the total points, each variable will occur exactly once in both arrays combined.
In test cases worth additional 20% of the total points, there will be only two variables, ‘x’ and ‘y’. It’s possible that the variables appear multiple times in both arrays.

SAMPLE TESTS

input

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

output

```
DA
```

input

```
4
4 5 iks ipsilon
1 iks 3 iks
```

output

```
NE
```

input

```
5
x 3 x y 3
x y 2 z 3
```

output

```
DA
```

Clarification of the third sample test:
By introducing substitutions x = 2, y = 3, z = 3, both arrays will become equal (2 3 2 3 3).

After spending most of his money on various projects, Nadan decided to afford high quality shoes for his software developers. Luckily for Nadan, he found *N* left shoes and *M* right shoes in his basement. Since their origin is unknown, the shoes come in various sizes.

Nadan asked you to pair as many shoes as possible (it’s important that a new pair cannot be selected after pairing all the shoes). Each pair should consist of one left shoe and one right shoe. While pairing the shoes, you should make sure that the ugliness is minimized. The ugliness of one pairing is defined as the maximal absolute difference of the shoe sizes between all pairs of shoes.

INPUT

The first line contains two positive integers *N* and M ($1 \le N$, $M \le 100\ 000$), the number of left shoes and right shoes, in that order.
The second line contains *N* numbers L_i ($1 \le L_i \le 10^9$), the sizes of the left shoes.
The third line contains *M* numbers R_i ($1 \le R_i \le 10^9$), the sizes of the right shoes.

OUTPUT

Output the minimal ugliness between all possible shoe pairings.

SCORING

In test cases worth 20% of total points, it will hold that $N = M$.
In test cases worth additional 50% of total points, it will hold that N, $M \le 5\ 000$.

SAMPLE TESTS

input	**input**	**input**

2 3
2 3
1 2 3
	4 3
	2 39 41 45
	39 42 46
		5 5
		7 6 1 2 10
		9 11 6 3 12

output	**output**	**output**
0	1	4

Clarification of the second sample test:
Nadan has 4 left and 3 right shoes, maximal number of pairs that can be obtained is 3. One possible pairing is: **39 - 46**, 41 - 42, 45 - 39, but ugliness of such pairing is 7 because of the first pair.
Better pairing would be:
39 - 39, 41 - 42, 45 - 46, with ugliness being equal to 1, which is minimal possible ugliness that can be obtained.

Everyone is afraid of something. Someone is afraid of darkness, someone is afraid of heights, someone is afraid of Vinnie Jones (all of us are afraid of Vinnie Jones), someone is afraid of singing before eating something...

There are many fears, but the greatest among all for Mirko is choosing a land for planting strawberries. Mirko's land can be described as a matrix with *N* rows and *M* columns. Some of the fields in the matrix are suitable for planting strawberries and some are not – weeds grow there. Mirko is looking for rectangular parts of the land that are completely filled with fields suitable for strawberry planting. Those kind of rectangles are called suitable rectangles. Also, Mirko is interested in the *potential value* of all fields in the matrix. The potential value of each field in the matrix is defined as the number of suitable rectangles that contain that field.

Since Mirko has troubles facing his fears, he asks you to only calculate the sum of all the fields' potential values.

INPUT

The first line contains two positive integers *N* and *M* (1 ≤ *N*, *M* ≤ 2 000), dimensions of the land.
The next *N* lines contains *M* characters each, representing the landscape. Each character can be either a '.' (dot) which represents a field suitable for planting or a '#' which represents weeds.

OUTPUT

Output the sum of all potential values of the matrix's fields.

SCORING

In test cases worth 20% of the total points, it will hold that 1 ≤ *N*, *M* ≤ 10.
In test cases worth additional 30% of the total points, it will hold that 1 ≤ *N*, *M* ≤ 300.

SAMPLE TESTS

input

```
2 3
.#.
..#
```

output

```
8
```

input

```
3 3
...
...
...
```

output

```
100
```

input

```
3 4
..#.
#...
...#
```

output

```
40
```

Clarification of the first sample test:
The following matrix describes the potential values of the land's fields. The sum of all potential values is 8.

2	**0**	**1**
3	**2**	**0**

Little Alan was bored so he asked Goran to give him an interesting problem. Since he's busy with preparing for exams, Goran could only recall one huge bipartite graph from his old days as a programming competitor. He gave the graph to Alan and said: *You have to colour the edges of this bipartite graph using as few colours as possible in such a way that there are no two edges of the same colour sharing a node.*

Alan excitedly ran to his room, took out his movable read/write device for its tape and start to work on the problem. However, he soon realized that he's missing something so he got back to Goran and said: *Give me an infinite tape and I will solve your problem!* Goran gave him a significant look: *Infinite tape? If you continue to theorize about everything, there won't be a single thing named after you.*

After seeing Alan starting to tear up, Goran decided to show mercy: *I will make it a bit easier for you. Let C be the smallest number of colours needed to paint the graph in the described way. I will let you use at most X colours, where X is the lowest power of 2 not less than C.*

Help Alan solve the problem.

Note: A bipartite graph is a graph whose nodes can be divided in two sets (or sides) in such a way that each edge of graph connects one node from the first set with one node from the second set.

INPUT

The first line contains three positive integers: L, R and M ($1 \le L, R \le 100\ 000$, $1 \le M \le 500\ 000$), representing the number of nodes in one side of the bipartite graph, number of nodes in the other side of the bipartite graph and the number of edges, in that order.
M lines follow, each containing two positive integers a_i ($1 \le a_i \le L$) and b_i ($1 \le b_i \le R$) which represent an edge between a_i-th node from the first side and b_i-th node from the second side of the bipartite graph. All pairs (a_i, b_i) will be unique.

OUTPUT

In the first line output a single positive integer K, the number of colours used.
In the next M lines output a single positive integer c_i ($1 \le c_i \le K$), label of the colour of the i-th edge.

SCORING

In test cases worth 20% of total points, it will hold that $L, R \le 100$.
In test cases worth additional 20% of total points, it will hold that $L, R \le 5\ 000$.

SAMPLE TESTS

input

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

output

```
2
1
2
1
2
1
```

input

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

output

```
4
1
2
3
4
```

Clarification of the second sample test:
Minimal number of colours is equal to 3. However, using 4 colours is also acceptable because that’s the lowest power of 2 which is not less than 3.