

Tasks

task	radio	crtalo	bard	buba
source code	radio.pas radio.c radio.cpp	crtalo.pas crtalo.c crtalo.cpp	bard.pas bard.c bard.cpp	buba.pas buba.c buba.cpp
input	standard input			
output	standard output			
time limit	1 second			
memory limit	32 MB (heap) + 8 MB (stack)			
points	30	45	55	70
	200			

RADIO

Slavko is driving his new Karambol GT20 car with a top-of-the-line radio (which, of course, Slavko made himself in his basement, next to the laundry washer).

Slavko's radio is very hi-tech and is equipped with two buttons that are used to manually change the frequency, and several buttons that change the frequency to one of the predefined frequencies. More precisely:

- The first button increases the frequency by 1 (one) MHz;
- The second button decreases the frequency by 1 (one) MHz;
- Each of the remaining N buttons changes the frequency to one of the N predefined frequencies.

Your job is to write a program that, given two frequencies A and B , computes the smallest number of button presses needed to change the frequency from A to B .

Input

The first line contains two integers A and B ($1 \leq A, B < 1000$, $A \neq B$).

The second line of input contains the integer N ($1 \leq N \leq 5$).

The following N lines contain the predefined frequencies (one integer per line, each less than 1000).

Output

Output the smallest number of button presses needed to change the frequency from A to B .

Example test cases

input	input	input
100 15	88 17	64 120
1	3	1
15	18	567
	1	
output	42	output
1	output	56
	2	

Slavko's newest invention is a drawing machine. The machine has a pencil and can be programmed to draw various shapes. This particular one draws shapes on a piece of paper divided into $N \times N$ squares.

In the beginning, the paper is empty. The starting position of the pencil is in the upper-left corner.

Slavko uses a microprocessor (which he found at Mirko's village hall, next to the bag of sodium and an empty bucket) to send movement orders to the machine. The orders state in which direction the machine moves its pencil (up, down, left, right). As the pencil moves, it leaves a trail on the paper.

If a command would have the pencil go out of the paper boundaries, that order is ignored, and the pencil stays where it was.

Given a sequence of commands, output what the paper looks like after the machine has completed those commands.

We say that the pencil **went through a square horizontally** if it entered or exited the square while performing the **left** or **right** command. Similarly, we define that the pencil **went through the square vertically** if it entered or exited the square while performing either the **up** or **down** command.

Format the output as follows:

- The squares that the pencil never entered are displayed as dots ('.');
- The squares that the pencil only went through horizontally are displayed as minus signs ('-');
- The squares that the pencil only went through vertically are displayed as vertical lines ('|');
- The squares that the pencil went through both horizontally and vertically are displayed as plus signs ('+').

Input

The first line of input consists of one integer, N . ($2 \leq N \leq 10$), the paper size.

The second line contains a string of up to 250 characters. That string represents the orders that Slavko issued to the machine. The orders are: up ('U'), down ('D'), left ('L') and right ('R').

Output

Output the appearance of the paper as described in the task.

The output should consist of exactly N lines, and each line of N characters. The only characters allowed in the output are dots (ASCII value 46), minus signs (45), vertical lines (124) and plus signs (43).

Example test cases

input

5
DRDRRUU

output

|..|.
++.|.
.+-.+
.....
.....

input

4
RRRDDDDULL

output

---+
...|
.-.+
...|

input

5
RRDDLLUURRDDLLUUR

output

+-.+..
|.|..
+-.+..
.....
.....

BARD

Every evening villagers in a small village gather around a big fire and sing songs.

A prominent member of the community is the bard. Every evening if the bard is present he sings a brand **new song** that no villager has heard before, and **no other song** is sung that night. In the event that the bard is not present, other villagers sing without him and exchange **all songs that they know**.

Given the list of villagers present for E consecutive evenings, output all villagers that know **all** songs sung during that period.

Input

The first line of input contains an integer N, $1 \leq N \leq 100$, the number of villagers. The villagers are numbered 1 to N. Villager number 1 is the bard.

The second line contains an integer E, $1 \leq E \leq 50$, the number of evenings.

The next E lines contain the list of villagers present on each of the E evenings. Each line begins with a positive integer K, $2 \leq K \leq N$, the number of villagers present that evening, followed by K positive integers separated by spaces representing the villagers.

No villager will appear twice in one night and the bard will appear at least once across all nights.

Output

Output all villagers that know all songs, including the bard, one integer per line in ascending order.

Example test cases

input

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

output

```
1
2
4
```

input

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

output

```
1
2
6
8
```

input

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

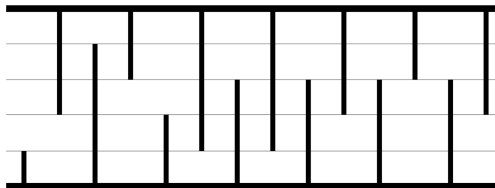
output

```
1
```

BUBA

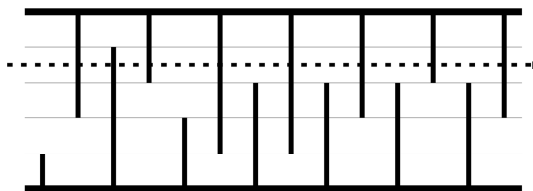
A Japanese firefly has flown into a cave full of obstacles: stalagmites (rising from the floor) and stalactites (hanging from the ceiling). The cave is N units long (where N is even) and H units high. The first obstacle is a stalagmite after which stalactites and stalagmites alternate.

Here is an example cave 14 units long and 5 units high (the image corresponds to the second example):



The Japanese firefly is not the type that would fly around the obstacle, instead it will choose a single height and ram from one end of the cave to the other **destroying all obstacles** in its path with its mighty kung-fu moves.

In the previous example, choosing the 4th level from the ground has the firefly destroying eight obstacles:



This is not the best choice because the firefly will end up less tired if it chooses either level one or five, as that would require destroying only seven obstacles.

You are given the width and length of the cave and the sizes of all obstacles.

Write a program that determines the **minimum number** of obstacles the firefly needs to destroy to reach the end of the cave, and on how many distinct levels it can achieve that number.

Input

The first line contains two integers N and H , $2 \leq N \leq 200\,000$, $2 \leq H \leq 500\,000$, the length and height of the cave. N will always be even.

The next N lines contain the sizes of the obstacles, one per line. All sizes are positive integers less than H .

Output

Output two integers separated by a single space on a single line. The first number is the minimum number of obstacles the firefly has to destroy and the second is the number of levels on which that can be achieved.

Example test cases

input

6 7
1
5
3
3
5
1

output

2 3

input

14 5
1
3
4
2
2
4
3
4
3
3
3
2
3
3
3

output

7 2