Maratona SBC de Programação 2025

This problem set is used in simultaneous contests:
Competencia Boliviana de Programación
The 2025 ICPC Gran Premio de Centroamerica
The 2025 ICPC Gran Premio de Mexico

September 13th, 2025

Warmup session

General Information

This problem set contains 4 problems; pages are numbered from 1 to 4, without considering this page. Please, make sure that your book is complete.

Read carefully the following instructions. Additional information, including time limits for problems, will be available in a document inside the contest manager system, BOCA.

A) Program name

1) Solutions written in C/C++ and Python, the filename of the source code is not significant, can be any name.
2) Solutions written in Java, filename should be: *problem_code*`.java` where *problem_code* is the uppercase letter that identifies the problem. Remember in Java the main class name and the filename must be the same.
3) Solutions written in Kotlin, filename should be: *problem_code*`.kt` where *problem_code* is the uppercase letter that identifies the problem. Remember in Kotlin the main class name and the filename must be the same.

B) Input

1) The input must be read from *standard input*.
2) The input is described using a number of lines that depends on the problem. No extra data appear in the input.
3) When a line of data contains several values, they are separated by *single* spaces. No other spaces appear in the input.
4) Every line, including the last one, ends with an end-of-line mark.
5) The end of the input matches the end of file.

C) Output

1) The output must be written to *standard output*.
2) When a line of results contains several values, they must be separated by *single* spaces. No other spaces should appear in the output.
3) Every line, including the last one, must end with an end-of-line.

Promotion:

Sociedade Brasileira de Computação

Problem A
Attention to the Meeting

Vinicius is at a board meeting of the “Instituto de Consultoria de Palestras e Comentários” (ICPC) thinking that it would be great if the board members were more concise and kept their speeches within the time allotted for each director, so that the meeting could end before lunch. Unfortunately, perhaps due to the nature of the institution, everyone loves to talk.

Knowing that

- there are N directors who will speak at the meeting;
- each director will speak for the same amount of time;
- and that between two consecutive speeches there is a 1-minute interval,

determine the maximum length of each speech, in minutes, so that the meeting lasts no more than K minutes.

Input

The first line contains an integer N ($1 \leq N \leq 100$), the number of directors. The second line contains an integer K ($1 \leq K \leq 1000$ and $K \geq N$), the maximum meeting duration in minutes. For all input cases, each director's speech lasts at least 1 minute.

Output

Your program should output a single line, containing a single integer, indicating the length of each board member's speech, in minutes.

Input example 1	**Output example 1**

`7`
`120`
	`16`

Explanation of sample 1:

There are 7 directors and the maximum meeting length is 120 minutes. If each director speaks for 16 minutes, we have $16 \times 7 = 112$ minutes. Since there are six breaks between speeches, and each break lasts one minute, we have 118 minutes in total. Note that, in this case, two minutes of the meeting time are not used, and that, if the speeches were longer than 16 minutes, the total time would exceed the 120-minute limit.

Input example 2	**Output example 2**

`1`
`10`
	`10`

Explanation of sample 2:

There is only one director and the meeting lasts 10 minutes. Therefore, the maximum speaking time of the director is 10 minutes.

Input example 3	**Output example 3**

`100`
`1000`
	`9`

Problem B
Interactive

This problem is interactive.

There is a hidden number N. Your goal is to discover N by asking at most 10 questions.

In each question, you must provide a number X. The system will respond whether X is less than, equal to, or greater than N.

Interaction

You can ask at most 10 questions to discover the hidden number N ($1 \leq N \leq 1000$). Each question must be asked in the format: "`?` X" (without quotes), where $1 \leq X \leq 1000$.

After each question, a line will be available for reading containing:

- `<`, if $N < X$.
- `=`, if $N = X$.
- `>`, if $N > X$.

When you are certain of the value of N, print "`!` N" (without quotes) and terminate the program.

The interactor is **not adaptive**, meaning the hidden number N is fixed throughout the entire interaction.

After each write to output, don't forget to *flush* the *buffer*. Otherwise, you may receive the verdict `TIME LIMIT EXCEEDED`. To do this, use:

- `fflush(stdout)` in `C`.
- `cout.flush()` in `C++`.
- `sys.stdout.flush()` in `Python`.
- `System.out.flush()` in `Java`.
- `System.out.flush()` in `Kotlin`.

Example of interaction 1

Read	Write
	`? 2`
`>`	
	`? 5`
`=`	
	`! 5`

Problem C
Bacon Number

Carlinhos loves movies, and recently he has been fascinated by the *Bacon Number*, which is defined as follows.

- The Bacon number of the actor Kevin Bacon is equal to 0;
- If the smallest Bacon number of an actor with whom X has appeared in the same movie is b, the bacon number of the actor X is $b + 1$.

That is, the Bacon number measures the shortest path between any actor and the actor Kevin Bacon, in which two actors are connected if they appeared together in the same movie.

Carlinhos is interested in a more general problem: given two actors, how to connect them through intermediate movies and actors? Given N movies, and, for each movie, which of the existing M actors acted in it. Carlinhos wants to answer Q queries: in the i-th of them, we want to compute some way to connect actor x_i with actor y_i. We must find some sequence $x_i = a_1, f_1, a_2, f_2, \ldots, f_{k-1}, a_k = y_i$, where $1 \leq a_j \leq N$ are actors and $1 \leq f_j \leq M$ are movies, and actor a_j acted in movies f_{j-1} and f_j, or indicate that no such sequence exists.

Input

In the first line of the input, two integers N ($1 \leq N \leq 100$) and M ($1 \leq M \leq 10^6$) are given, the number of movies and the number of actors.

N lines follow. In the i-th line, the first integer n_i ($1 \leq n_i \leq M$) denotes the number of actors in movie i. Next, n_i numbers in ascending order separated by spaces: the indices, from 1 to M, of the actors who acted in movie i.

The next line, contains an integer Q ($1 \leq Q \leq 10^4$): the number of queries.

The next Q lines describe the queries. In the i-th of them, read two numbers x_i, y_i ($1 \leq x_i \neq y_i \leq M$), the actors we want to connect. It is guaranteed that the total number of actors in the movies is at most 10^6. That is, $\sum_i n_i \leq 10^6$.

Output

For each of the queries, if there is no sequence, print a line with -1. Otherwise, print two lines. In the first line, print the number of actors k_i ($2 \leq k_i \leq 10^6$) in some way to connect x_i and y_i. In the second, print the sequence as described, with k_i actors and $k_i - 1$ movies, alternating. If there is more than one way to connect the actors, print any of them.

Input example 1	**Output example 1**

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

Problem D
Amusement Park Adventure

Meet Carlitos, a spirited adventure enthusiast with an insatiable love for amusement parks. Despite his vibrant passion, Carlitos faces a unique challenge – his height. As he eagerly plans his weekend escapade, he realizes that his vertical limitations might hinder his amusement park experience. It's not just about choosing a park; it's about finding one where he can enjoy the thrill of the rides.

Picture the kaleidoscope of colors, the jubilant laughter, and the heart-pounding rush of the rides. Carlitos has always been drawn to the energy of amusement parks. With the weekend approaching, he pores over park brochures, studying the height requirements of each ride. His goal is to maximize his enjoyment, and that's where you come in.

Your task is to help Carlitos determine the number of rides he can enjoy at a specific park. By considering his height and the minimum height requirements of each ride, guide him in making the most of his amusement park adventure.

Input

The first line contains two integers, N and H ($1 \leq N \leq 6$ and $90 \leq H \leq 200$), representing the number of rides in a park and Carlitos' height in centimeters, respectively.

The second line contains the minimum heights $A_1, \ldots, A_N$ ($90 \leq A_i \leq 200$) of each ride in the park.

Output

Output a single line with an integer indicating the number of rides Carlitos can go on, that is, the number of rides for which Carlitos' height is at least as large as the minimum height required.

Input example 1	**Output example 1**

`1 100`
`100`
	`1`

Input example 2	**Output example 2**

`6 120`
`200 90 100 123 120 169`
	`3`