Secret Boxes

Time limit: *10000 ms*
Memory limit: *256 MB*

⇄Interactive

Suzana is a very generous teacher. She wants to give her students some gifts for passing the exam. She has prepared N boxes which contain some candies for her students. Each box has a different number of candies ranging from 1 to N. There is exactly one box that contains k candies for each k from 1 to N.

Suzana wants to distribute the gifts to all students so that the better ranked student will get more candies than the worse ranked student. But, she forgot the number of candies in each box because the boxes look similar. Fortunately, she has a tool that can compare the weight between a pair of boxes with another pair of boxes. The tool can determine whether the total weight of the first pair is equal, less, or greater than the total weight of the second pair.

Interaction

First, you should read two integers, T and N, representing the number of test cases and the number of candy boxes for each case. You have to process T different cases for the same N in one test file.

Then, you can make some queries to compare two different pairs of boxes. To make a query, you can print the query in the following format:

```
? A B C D
```

where (A, B) is the first pair of boxes and (C, D) is the second pair of boxes. The boxes A, B, C, and D should be pairwise distinct.

After that, you should read a character representing the result of comparison:

- If the character is `<`, then it means the total weight of box A and box B is **less** than the total weight of box C and box D.
- If the character is `=`, then it means the total weight of box A and box B is **equal** to the total weight of box C and box D.
- If the character is `>`, then it means the total weight of box A and box B is **greater** than the total weight of box C and box D.

If you already know the number of candies in each box, then you can print your answer in the following format:

```
! candies[1] candies[2] ... candies[N]
```

where $candies[i]$ is the number of candies in the i-th box.

After that, you should read a string that represents whether your answer is correct or wrong. If the string is `OK`, it means your answer is correct. Otherwise, you will receive string `WRONG` which means your answer is wrong and you should terminate your program.

If your answer is correct, you can continue to process the next case. If you have answered correctly all the T cases, you can terminate your program. Please note that you will only get a score if you have answered correctly all the T cases in one test file.

Constraint and notes:

- $T = 5$
- $5 \leq N \leq 1500$
- The values of N for all test files are $5, 10, 20, 40, 80, 160, 320, 640, 1280, 1500$
- You can only ask at most $30\,000$ queries for one case (printing the number of candies is not counted as query), otherwise you will get `Too many queries` verdict.
- There is exactly one box so that $candies[i] = k$ for each k from 1 to N.

Additional notes on interaction:

- Don't forget to flush after every output operation!
- The judge is not adaptive, which means that number of candies for all boxes are already determined at the start of the program execution and will not change until the end.
- The number of candies in a specific case will never change. A solution will always be tested on a same set of cases. However, the order in which the judge presents the cases may be different in each evaluation. For example, when you submit a solution the first time, the judge may give case 1 before case 2. When you submit a second time the judge may give case 2 before case 1.
- The sample test has $T = 2, N = 5$ only for an exemplification of the interaction between a solution and the judge program. The example queries only showcase the output format, and should not be considered successful identifications of the number of candies.
- The "Run Input" in the web IDE does not allow you to interact with a real judge program. It only sends the provided input to your program non-interactively. We recommend testing your solution's interaction using "Run Examples", which will interact with a real judge program on the $T = 2, N = 5$ sample test.

Interaction		Explanation
`2 5`		This is an example sequence of interaction. There are two test cases. Note that the example queries only showcase how to interact with the judge program. They are not necessarily sufficient to solve the test cases.
	`? 1 4 3 2`	
`=`		
	`? 2 3 4 5`	
`<`		
	`? 5 3 1 2`	
`>`		
	`! 1 2 3 4 5`	
`OK`		
	`? 1 2 3 4`	
`>`		
	`! 5 4 3 2 1`	
`OK`		

WORKSPACE / SUBMIT