SAC '22 Code Challenge 4 P5 - Obligatory Interactive Problem

Time limit: 5.0s **Memory limit:** 256M
Python: 10.0s

You want to determine the heights of N students (H, which is a permutation from 1 to N), but you do not have a measuring stick.

Instead, you can ask up to $Q_1 + Q_2$ queries of 2 types:

`c i j`: Query the index of the taller person (i.e., if $H_i > H_j$, it will return i; if $H_i < H_j$, it will return j).

`h k`: Query the height of the k^{th} student.

You can ask up to Q_1 `c` queries and Q_2 `h` queries.

Once you have the permutation, output `!` followed by N space-separated integers, representing the height of each student.

Can you recover their heights?

Constraints

$1 \leq i, j, k \leq N$

$i \neq j$

Subtask 1 [20%]

$3 \leq N \leq 500$

$Q_1 = 300\,000, Q_2 = 300$

Subtask 2 [5%]

$3 \leq N \leq 500$

$Q_1 = 150\,000, Q_2 = 300$

Subtask 3 [50%]

$3 \leq N \leq 10\,000$

$Q_1 = 300\,000, Q_2 = 6\,000$

Subtask 4 [25%]

$3 \leq N \leq 10\,000$

$Q_1 = 100\,000, Q_2 = 6\,000$

Interaction

This is an interactive problem. The first line of input contains N, Q_1, Q_2, and T, representing the number of students, the number of `c` queries you can make, the number of `h` queries you can make, and the subtask of this test. Then, you will make at most $Q_1 + Q_2$ queries and output the heights.

If you believe you have the correct list of heights, you may output `!` followed by a space-separated list of N integers (followed by a `\n` character), representing your guess of the list of capacities in order from the first student to the last. You must terminate your program immediately after performing this operation. Note that this operation does **not** count towards the limit of $Q_1 + Q_2$ operations.

The grader is not adaptive.

If at any point your query is malformed or you exceed the number of available queries, your program will receive `-1`, and the interactor will terminate. Upon receiving `-1`, your program should terminate to avoid an undefined verdict.

If you attempt an invalid operation (such as invalid output format), you will receive a `Presentation Error` verdict.

If you exceed the available query limit, query incorrectly, or output incorrect final heights, you will receive a `Wrong Answer` verdict.

Please note that you may need to flush `stdout` after each operation, or interaction may halt. In C++, this can be done with `fflush(stdout)` or `cout << flush` (depending on whether you use `printf` or `cout`). In Java, this can be done with `System.out.flush()`. In Python, you can use `sys.stdout.flush()`.

Sample Interaction

`>>>` denotes your output. Do not print this out.

The heights are $[5, 4, 2, 1, 3]$.

```
5 300000 300 1
>>> c 4 3
3
>>> c 3 5
5
>>> c 5 2
2
>>> c 2 1
1
>>> ! 5 4 2 1 3
```

Sample Explanation

Since each comparison is strictly increasing, you know that the first query was a 1 and progressively increases until 5. Additionally, this interaction outputs the correct heights and uses fewer `c` queries than the maximum given (Q_1), so this solution will receive an `Accepted` verdict.