Flying orcs

Contrary to some stories telling otherwise—mostly spread by the *Association of Middle-earth Gardeners*—Samwise Gamgee wasn't the one to save Frodo from the Tower of Cirith Ungol...

Frodo is in danger (again) and hence it is up to Aragorn, Legolas and Gimli to save the hobbit caught in Mordor. In lack of a guide talking to themselves this means that they'll have to find their way cross country through enemy territory. However, since one of the main points of their plan is built around getting to the tower *unseen*, they face the issue of evading Sauron's (hopefully not actually all-seeing) Eye.

In order to do so they want to identify the area under Sauron's watch first. To this end, Legolas (who seems to be quite orderly) already has a map of Mordor at hand, which is composed of m columns and n rows of *grid squares*. We have:

- Each grid square is either totally in Sauron's field of vision or completely outside of it.
- The area under Sauron's watch is completely contained in the mapped area.
- Our three friends know a point (Mount Doom) under Sauron's watch.
- The area under Sauron's watch is connected and contains no holes.

Here we call two grid squares neighbouring when they share an edge; *connected* means that any two grid squares watched by Sauron can be connected by a sequence of neighbouring grid squares again under Sauron's watch; conversely *no holes* means that any field *not* under Sauron's watch can likewise be connected to the border of the map by a sequence of neighbouring fields not watched by Sauron.

Fortunately, there are some orcs and a trebuchet (a huge catapult) left from the Battle of the Pelennor Fields. Legolas suggests throwing orcs at prescribed points and checking the Eye's reactions to find out whether the respective grid square is watched by Sauron (an earlier suggestion to just throw Gimli was rejected—accompanied by loud bursts of dwarven curses). As time is running out, they want to use as few orcs as possible. Help them by writing a program selecting the target points for them!

Interaction

This is an interactive task. In the beginning you can read the two integers m and n separated by spaces from standard input. In the same way, the next line contains the coordinates x, y of a point watched by Sauron. We have $1 \leq x \leq m$ and $1 \leq y \leq n$.

Following that you can send queries to the grader program. Each of these should have the form `1` x y where x, y give the target coordinates for the trebuchet in the same format as above. As a response you can read a single integer from standard input: `1` if the corresponding field is watched by Sauron and `0` otherwise.

Once you're sure about the area under Sauron's watch, output a single line consisting of the integer `2` and then n lines of m space separated integers each. The i-th number in the j-th row should be `1` if Sauron watches grid square (i, j) and `0` otherwise.

Important notice

In order to ensure that communication between the two programs works, you have to flush standard output after each of your program's outputs. When using *cout* it suffices to append $\ll$ *flush* to each output, e.g.

$$\textit{cout} \ll \texttt{"1 5 1\textbackslash n"} \ll \textit{flush};$$

or to always use *endl* for line breaks. When using *printf* you have to call *fflush*(*stdout*) after each output.

Constraints and grading

In all testcases we have $1 \leq m, n \leq 500$.

Subtask 1 (40 points). $m \leq 200$ and $n \leq 250$

Subtask 2 (40 points). $m, n \leq 400$

Subtask 3 (20 points). No further constraints.

In the grading process we compare your program with our (very efficient) sample solution. The score for a given subtask is computed as follows: for each testcase you must not issue more than 250 000 queries (note that this is actually not a severe restriction as it still allows you to query every single grid square). If you use more queries, do not use the above format for queries, or if your program outputs the wrong answer, you get 0 points for the whole subtask. Otherwise, we form for each testcase in this subtask the ratio

$$\alpha := \frac{\text{orcs thrown by your program}}{\text{orcs thrown by our sample solution}}$$

and then consider the maximum $\alpha_{\max}$ of these. The following table tells you how many points you will get based on this value:

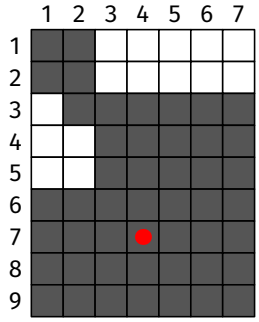
$\alpha_{\max}$	$[0, 2]$	$(2, 5]$	$(5, 20]$	$(20, 50]$	$(50, \infty)$
Fraction of points	100%	75%	50%	25%	0%

Please note that we have chosen the non-sample testcases in such a way that

- ... a solution that simply queries all grid squares in each testcase won't get any points.
- ... the number of queries needed is rather large (so additive constants have virtually no effect on grading).

Sample case

The following picture represents a valid instance:



Note however that it would *not* be valid if the square $2, 3$ were excluded (then the area wouldn't be connected according to our definition). Likewise it wouldn't be valid for example if the square $3, 6$ were excluded (then the area would contain a hole).

A possible interaction could read as follows:

Your program	Grader	Explanation
	`7 9`	dimensions of the map
	`4 7`	a square known to be watched by Sauron
`1 1 1`		querying grid square 1, 1
	`1`	it is watched by Sauron
`1 5 1`		querying grid square 5, 1
	`0`	it is not watched by Sauron
`2`		your program wants to output the answer

`1 1 0 0 0 0 0`
`1 1 0 0 0 0 0`
`0 1 1 1 1 1 1`
`0 0 1 1 1 1 1`
`0 0 1 1 1 1 1`
`1 1 1 1 1 1 1`
`1 1 1 1 1 1 1`
`1 1 1 1 1 1 1`
`1 1 1 1 1 1 1`		

		your solution is correct and gets accepted

Note that of course the above queries do not suffice to actually identify the area in question. Our sample solution for example uses 40 queries.

Limits

Time: 2 s
Memory: 1024 MiB

The above time limit also contains the runtime of our grader and in particular the time needed for the communication between the two programs. We guarantee however, that it is possible in C++ to issue 250 000 queries and output the solution in less than 2 seconds.

Feedback

restricted feedback is given for this task, i.e. the score shown equals your real score on this submission. However, for each testcase group you are only shown the verdict for the first testcase with minimal score in that group. (The order of the testcases in each of the groups is fixed.)