Find the box

Problem Name	Find the Box
Time Limit	1 seconden
Memory Limit	1 gigabyte

Maj werkt als robotonderzoeker aan Lund University. Ze heeft een een waardevolle schat in de kelder van de universiteit ontdekt. De schat is in een doos in een lege kamer onder de grond. Jammergenoeg, kan Maj niet gewoon naar de doos gaan zoeken. Het is erg donker in de kelder en het zou argwaan wekken om daarheen te gaan met een zaklamp. De enige manier om de schat te zoeken is door een robotstofzuiger te besturen die in de kelder staat.

De kelder wordt gerepresenteerd door een $H \times W$ grid, met rijen genummerd van 0 tot $H-1$ (van boven naar beneden) en de kolommen genummerd van 0 tot $W-1$ (van links naar rechts). Dus linksboven is $(0,0)$ en rechtonder is $(H-1,W-1)$. De doos met de schat ligt op een onbekende plek maar niet op positie $(0,0)$. De robotstofzuiger start iedere nacht linksboven en beweegt door de kelder.

Iedere nacht kan Maj de robot een serie instructies geven over hoe die door de ruimte moet bewegen met een string die bestaat uit de karakters "<", ">", "^" en "v". In andere woorden, als de robot op (r,c) staat en alle kanten zijn niet-geblokkeerd, "<" beweegt de robot eentje naar links $(r,c-1)$, ">" beweegt de robot eentje naar rechts $(r,c+1)$, "^" beweegt de robot eentje omhoog $(r-1,c)$, en "v" beweegt de robot eentje naar beneden $(r+1,c)$.

De muren van de kelder zijn hard, dus als de robot probeert buiten het grid te komen, gebeurt er niets. De doos is ook hard, en kan niet geduwd worden. Aan het eind van iedere nacht, rapporteert de robot zijn locatie en dan gaat hij terug naar de hoek linksboven.

Tijd is belangrijk, dus Maj wil de doos in zo weinig mogelijk nachten vinden.

Interaction

Dit is een interactief probleem.

- Je programma leest één regel in met twee getallen H en W: de hoogte en de breedte van het grid.

- Daarna begint de interactie met de grader. In elke ronde van de interactie geef je een opdracht door een vraagteken "`?`" te printen gevolgd door een niet-lege string s met de instructies deze bestaan uit de karakters "`<`", "`>`", "`^`", "`v`". De lengte van deze string is maximaal $20\,000$. Daarna leest je programma twee getallen in r, c ($0 \leq r \leq H-1$, $0 \leq c \leq W-1$), de locatie van de robot na het opvolgen van de instructies. Let op, de robot gaat altijd terug naar $(0, 0)$ na iedere opdracht.
- Zodra je de locatie van de doos weet, print "`!`" gevolgd door twee getallen r_b, c_b, de rij en de kolom van de doos ($0 \leq r_b \leq H-1$, $0 \leq c_b \leq W-1$). Hierna moet je programma stoppen zonder nieuwe opdrachten te geven. Dit laatste statement telt niet als opdracht voor het vaststellen van je score.

Make sure to flush standard output after issuing a query, or else your program might get judged as Time Limit Exceeded. In Python, `print()` flushes automatically. In C++, `cout << endl;` also flushes in addition to printing a newline; if using printf, use `fflush(stdout)`.

De grader is non-adaptive, dit betekent dat de positie van de box voor de interactie begint al is bepaald.

Constraints and Scoring

- $1 \leq H, W \leq 50$.
- De doos staat nooit op $(0, 0)$. Dat betekent dat $H + W \geq 3$.
- Elke opdracht kan maximaal $20\,000$ instructies hebben.
- Je kan maximaal $2\,500$ opdrachten geven (het printen van het antwoord telt niet als een opdracht).

Je oplossing wordt getest op een aantal testcases. Als je oplossing faalt op een van deze testcases (bijvoorbeeld door het geven van een verkeerde positie (Wrong Answer), een run time error (Runtime Error), exceeding the time limit (Time Limit Exceeded), etc), krijg je 0 punten en de bijbehorende grader uitkomst.

Als je programma in *alle* testcases de positie van de doos vindt, krijg je de grader uitkomst Accepted, en je score wordt als volgt berekend:

$$\text{score} = \min\left(\frac{100\sqrt{2}}{\sqrt{Q}}, 100\right) \text{ points},$$

waar Q het maximum aantal opdrachten is per testcase is. Het printen van het antwoord telt niet als opdracht. De score wordt afgerond naar het dichtsbijzijnde gehele getal.

Om 100 punten te krijgen, mag je programma maximaal $Q = 2$ vragen gebruiken. De tabel hieronder geeft sommige waarden van Q en de bijbehorende scores.

Q	2	3	4	5	...	20	...	50	...	2500
Score	100	82	71	63	...	32	...	20	...	3

Testing Tool

To facilitate the testing of your solution, we provide a simple tool that you can download. See “attachments” at the bottom of the Kattis problem page. The tool is optional to use, and you are allowed to change it. Note that the official grader program on Kattis is different from the testing tool.

Example usage (with $H = 4$, $W = 5$, and the hidden box at position $r = 2$, $c = 3$):

For Python programs, say `solution.py` (normally run as `pypy3 solution.py`):

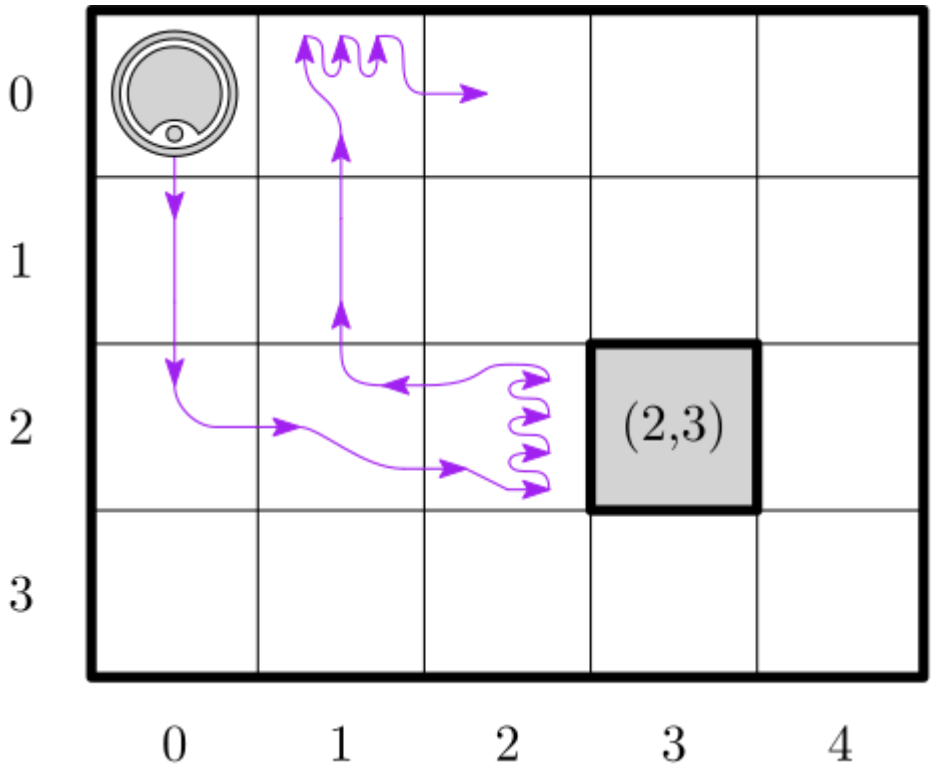
```
python3 testing_tool.py pypy3 solution.py <<<"4 5 2 3"
```

For C++ programs, first compile it (e.g. with `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`) and then run:

```
python3 testing_tool.py ./solution.out <<<"4 5 2 3"
```

Example

Hier is een voorbeeld testcase. Het grid heeft hoogte $H = 4$ en breedte $W = 5$, en de doos is op positie $(r, c) = (2, 3)$. De figuur hieronder laat het pad van de robot zijn wanneer hij de volgende opdracht krijgt “`? vv>>>>>><^^^^>`”, waarbij de robot eindigt op positie $(r, c) = (0, 2)$. Voor de start van de tweede opdracht, gaat de robot terug naar de hoek linksboven $(0, 0)$. Daarna volgt de tweede opdracht “`? >>>>>>>>vvvvvvvvvv`” waarbij de robot in de hoek rechtsonder eindigt $(r, c) = (3, 4)$. Nu gokt de oplossing het antwoord “`! 2 3`”, en dat is de juiste positie van de doos.



grader output	your output
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3