箱を探せ (Find the Box)

問題名	箱を探せ (Find the Box)
実行時間制限	1 sec
メモリ制限	1 GB

Maj はルンド大学で働くロボットの研究者である．彼女は大学の地下室にある宝物について知った．その宝物は，地下深くの空き部屋にある箱の中にある．しかし，Maj はそこに行って箱を探すということはできない．地下室の中はとても暗く，ライトを使うと疑われてしまうかもしれないためである．宝を探すために彼女に残された道は，地下室を担当しているロボットの掃除機をリモートでコントロールすることである．

地下室は $H \times W$ のグリッドとして表され，行は上から下の順に 0 から $H-1$ までの番号で表され，列は左から右の順に 0 から $W-1$ の番号で表される．したがって，左上のマスは $(0,0)$ であり，右下のマスは $(H-1, W-1)$ である．

宝物の入った箱は $(0,0)$ でないある未知のマスにある．毎晩，ロボットの掃除機は左上の隅のマスから始めて地下室の中を動き回る．

各晩に，Maj はロボットの動き方を “<”, “>”, “^” , “v” からなる文字列によって指定することができる．より正確には，ロボットがマス (r,c) にいるとき，もしロボットの動きが妨げられない場合には，“<” によって左のマス $(r, c-1)$, “>” によって右のマス $(r, c+1)$, “^” によって上のマス $(r-1, c)$, “v” によって下のマス $(r+1, c)$ に動く．

地下室の壁は固定されているため，ロボットがグリッドの外へ移動しようとした場合は何も起こらない．また箱も固定されているため，ロボットがそのマスに移動しようとした場合，箱が押しのけられることはない。それぞれの晩に，ロボットは最終的な場所を報告し，左上の隅に戻る．

時間が惜しいため，Maj はできるだけ少ない晩で箱を見付けたい．

インタラクション

この問題はインタラクティブな問題である．

- あなたのプログラムは 1 行に 2 つの整数 H と W を読み込む．それぞれグリッドの縦の長さと横の長さを表す．

- その後，あなたのプログラムは採点プログラムとやりとりを行う．それぞれのやりとりのラウンドでは，“?” に続いて “<”, “>”, “^”, “v” の文字からなる空でない文字列 s を 出力せよ．文字列の長さは $20\,000$ 以下でなければならない．そして，あなたのプログラムは 2 つの整数 r, c $(0 \leq r \leq H-1, 0 \leq c \leq W-1)$ を読み込まなければならない．この 2 つの整数はその晩の指示をすべて実行した後のロボットの場所を表す．ロボットは各晩の指令の後にマス $(0, 0)$ へ戻ることに注意せよ．

- 箱の位置が分かったときには，“!” に続いて 2 つの整数 r_b, c_b $(0 \leq r_b \leq H-1, 0 \leq c_b \leq W-1)$ を出力せよ．それぞれ箱のあるマスの行・列の番号を表す．この後，あなたのプログラムは追加のクエリを呼び出すことなく終了しなければならない．また，この最後の出力は得点を決める際のクエリの回数にはカウントされない．

各クエリの後には必ず標準出力を flush せよ．さもなくば，あなたのプログラムは Time Limit Exceeded と判定される可能性がある．Pythonでは，`print()` によって自動的に flush される．C++ では，`cout << endl;` によって，新しい行が出力されるとともに flush がされる．もし printf を使う場合は，`fflush(stdout)` を用いよ．

採点プログラムはアダプティブではない．すなわち，箱の位置はやりとりが始まる前に既に決められている．

制約と評価方法

- $1 \leq H, W \leq 50$.
- 箱は $(0, 0)$ に位置しない．よって $H + W \geq 3$ である．
- それぞれのクエリの指令の長さは $20\,000$ 以下でなければならない．
- クエリの回数は $2\,500$ 以下でなければならない．

あなたのプログラムはいくつかのテストケースに対してテストされる．もしあなたのプログラムがいずれかのケースで失敗した場合 (間違った箱の位置を答えた場合 (Wrong Answer)，実行時エラー (Runtime Error)，実行時間制限超過 (Time Limit Exceeded) など)，得点は 0 となり，適切な判定が下される．

もしあなたのプログラムがすべてのテストケースについて箱の位置を正しく答えた場合，判定は Accepted となり，得点は以下のように計算される:

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

ただし，Q はすべてのテストケースに対する最大のクエリの回数である．最後の答えの出力はクエリの回数にカウントされない．得点は最も近い整数に丸められる．

特に，100 点を得るためには，あなたのプログラムはすべてのテストケースに対して $Q = 2$ 以下のクエリで正解する必要がある．下の表はいくつかの Q の値とそれに対応する得点である．

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

テストのためのツール

あなたが解法をテストすることを容易にするため，シンプルなツールがダウンロードできるようになっている．Kattis の問題ページの下部の "attachments" を見よ．このツールを必ずしも使う必要はなく，また変更することも許される．ただし，Kattis の実際の採点プログラムとこのツールは異なることに注意せよ．

使い方の例 ($H = 4, W = 5$ であり，隠された箱の位置が $r = 2, c = 3$ であるとき):

Python のプログラムの場合，例えば `solution.py` の場合，(通常は `pypy3 solution.py` とすれば実行されるが) 以下を実行せよ:

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

C++ のプログラムの場合，まずコンパイルして (例えば `g++ -std=gnu++17 solution.cpp -o solution.out` と打って) その後以下を実行せよ:

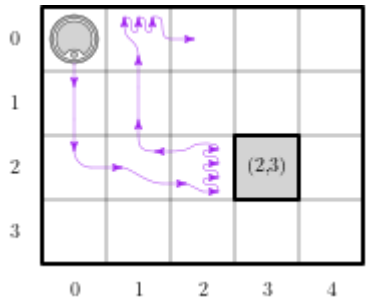
```
python3 testing_tool.py ./solution.out <<<"4 5 2 3"
```

例

サンプルのテストケースについて考える．グリッドは縦の長さ $H = 4$，横の長さ $W = 5$ であり，箱の位置は $(r, c) = (2, 3)$ である．下の図は 1 つ目のクエリの指示 "`? vv>>>>>><^^^^>`" に従ったときのロボットの移動経路を表す．最終的に，ロボットはマス $(r, c) = (0, 2)$ に到達する．

2 つ目のクエリの前に，ロボットは左上の隅のマス $(0, 0)$ に再び移動する．そしてあなたのプログラムは別のクエリ "`? >>>>>>>>vvvvvvvvv`" によって指示を行い，最終的にロボットはマス $(r, c) = (3, 4)$ に到達する．

最後にあなたのプログラムは箱の位置を推測し，"`! 2 3`" と出力する．これは正しい箱の位置である．



採点プログラムの出力	あなたのプログラムの出力
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3