Find the Box

Aufgabenname	Find die Box
Time Limit	1 Sekunde
Memory Limit	1 Gigabyte

Maj ist eine Robotertechnikforscherin, die an der Lund University arbeitet. Sie hat von einem wertvollen Schatz im Keller der Universität gehört. Der Schatz befindet sich in einem leeren Raum tief im Untergrund. Unglücklicherweise kann Maj nicht einfach gehen und nach der Box suchen. Es ist im Keller sehr dunkel und Gehen mit Licht würde verdächtig erscheinen. Ihre einzige Möglichkeit, den Schatz zu finden, ist das Fernsteuern eines Roboterstaubsaugers, der sich im Keller befindet.

Der Keller wird durch ein $H \times W$ Gitter dargestellt. Die Reihen werden von 0 bis $H-1$ nummeriert (von oben nach unten). Die Spalten sind von 0 bis $W-1$ nummeriert (von links nach rechts). Dies bedeutet, dass die oberste linke Zelle $(0,0)$ ist und die unterste rechte Zelle bei $(H-1, W-1)$ liegt.

Die Schatztruhe befindet sich in einer unbekannten Zelle, aber nicht in der Zelle $(0,0)$.

Jede Nacht startet der Saugroboter links oben und bewegt sich im Keller umher.

In jeder Nacht kann Maj dem Roboter eine Folge von Anweisungen geben, wie er sich bewegen soll. Die Anweisungen bestehen jeweils aus einem String mit den Zeichen "<", ">", "^" und "v". Formal bedeutet das: Wenn der Roboter in der Zelle (r,c) steht und in keine Richtung blockiert ist, dann bewegt "<" den Roboter nach links zur Zelle $(r, c-1)$, ">" bewegt den Roboter nach rechts zur Zelle $(r, c+1)$, "^" bewegt den Roboter nach oben zur Zelle $(r-1, c)$ und "v" bewegt den Roboter nach unten zur Zelle $(r+1, c)$.

Der Keller hat feste Wände. Wenn der Roboter versucht sich nach aussen zu bewegen, dann passiert nichts. Die Box ist ebenfalls stabil und kann nicht verschoben werden. Am Ende jeder Nacht meldet der Roboter seine Position und bewegt sich nach links oben zurück.

Wesentlich ist die Zeit, sodass Maj entscheidet, die Box in möglichst wenig Nächten zu finden.

Interaktion

Dies ist eine interaktive Aufgabe.

- Dein Programm sollte mit dem Einlesen von einer Zeile mit zwei ganzen Zahlen H und W beginnen: der Höhe und Breite des Gitters.
- Dann sollte dein Programm mit dem Grader kommunizieren. In jeder Interaktionsrunde solltest du ein Fragezeichen "`?`" gefolgt von einem nicht leeren String s drucken, der die Zeichen "`<`", "`>`", "`^`", "`v`" enthält. Die Länge des Strings beträgt maximal $20\,000$. Dann sollte dein Programm zwei ganze Zahlen r, c ($0 \leq r \leq H-1$, $0 \leq c \leq W-1$) einlesen, nämlich die Position des Roboters nach dem Ausführen der Anweisungen. Beachte, dass der Roboter nach jedem Befehl zu seiner Ausgangsposition $(0, 0)$ zurückgeht.
- Wenn du die Position der Box kennst, gib "`!`" gefolgt von zwei ganzen Zahlen r_b, c_b aus, der Reihe und Spalte der Box ($0 \leq r_b \leq H-1$, $0 \leq c_b \leq W-1$). Dann muss dein Programm ohne weitere Anweisungen enden. Die letzte Ausgabe zählt nicht als Anweisung für die Berechnung deiner Punkte.

Stelle sicher, dass der Standard Output nach einer Anweisung geleert wird (flush standard output), sonst wird dein Programm mit "Time Limit Exceeded" bewertet.

In Python flusht `print()` automatisch. In C++ flusht `cout << endl;` in Verbindung mit Ausgeben einer neuen Zeile automatisch. Wenn printf verwendet wird, schreibe `fflush(stdout)`.

Der Grader ist nicht-adaptiv, das bedeutet, dass die Position der Box vor dem Beginn der Interaktion eindeutig bestimmt ist.

Beschränkungen und Bewertung

- $1 \leq H, W \leq 50$.
- Die Box ist nie bei $(0, 0)$ platziert. Das bedeutet, dass $H + W \geq 3$.
- Jede Anfrage besteht aus höchstens $20\,000$ Instruktionen.
- Du kannst höchstens $2\,500$ Anfragen stellen (die finale Antwort zählt dabei nicht als Anfrage).

Deine Lösung wird auf mehreren Testfällen getestet. Wenn deine Lösung bei *einem* dieser Testfälle versagt (z.B. durch Melden einer falschen Boxposition (WA), Absturz (RTE), Überschreiten des Zeitlimits (TLE), etc.), bekommst du 0 Punkte und die entsprechende Rückmeldung.

Wenn dein Programm die Position der Box erfolgreich in *allen* Testfällen findet, dann bekommst du die Rückmeldung Accepted und die Punktezahl wird folgendermassen berechnet:

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

wobei Q die *maximale* Anzahl von Abfragen ist, die bei irgendeinem Testfall verwendet wird. Die letzte Antwort zählt nicht als Anfrage. Die Punktezahl wird auf die nächste ganze Zahl gerundet.

Insbesondere, um 100 Punkte zu erreichen, muss dein Programm jeden Testfall mit höchstens $Q = 2$ Abfragen lösen. Die Tabelle darunter zeigt einige Werte von Q und die zugeordnete Punktezahl.

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

Test-Tool

Um das Testen deiner Lösung zu ermöglichen, stellen wir ein einfaches Tool zum Download zur Verfügung. Siehe “attachments” unten auf der Kattis Aufgabenseite. Das Tool kann optional verwendet werden und du darfst es verändern. Beachte, dass der offizielle Grader auf Kattis sich vom Test-Tool unterscheidet.

Ein Beispiel für die Verwendung (mit $H = 4$, $W = 5$, und versteckter Box auf Position $r = 2$, $c = 3$):

Für Python Programme: rufe `solution.py` auf (läuft normalerweise als `pypy3 solution.py`):

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

Für C++ Programme, kompiliere es zuerst (z.B. mit `g++ -std=gnu++17 solution.cpp -o solution.out`) und rufe auf:

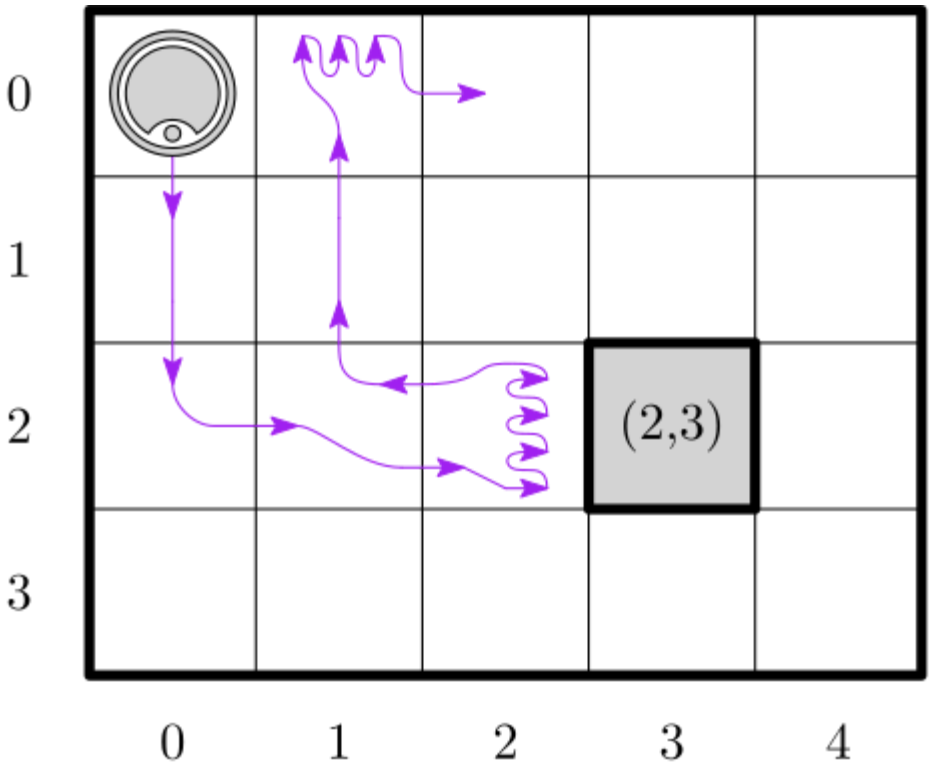
```
python3 testing_tool.py ./solution.out <<<"4 5 2 3"
```

Beispiel

Nimm den Beispielfall: Das Gitter hat die Höhe $H = 4$ und Breite $W = 5$, und die Box befindet sich an der Position $(r, c) = (2, 3)$.

Die Zeichnung darunter zeigt den Roboterpfad bei folgenden Anweisungen in der ersten Abfrage “`? vv>>>>>><^^^^>`” mit dem Ergebnis der Roboterposition bei $(r, c) = (0, 2)$.

Vor der zweiten Abfrage geht der Roboter wieder zur Position $(0, 0)$. Die Lösung stellt eine andere Abfrage “`? >>>>>>>vvvvvvvvv`”, für die der Roboter an der untersten rechten Ecke $(r, c) = (3, 4)$ stehen bleibt. Nun entscheidet sich das Programm für die Antwort, indem es “`! 2 3`” schreibt, was die richtige Position der Box ist.



Grader Output	Dein Output
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3