Find the Box

Nome problema	Find the box
Limite di tempo	1 secondi
Limite di memoria	1 gigaottetti

Maj è un ricercatore di robotica che lavora presso l'università di Lund. Ha saputo di un prezioso tesoro nella cantina dell'università. Il tesoro è in una scatola situata in una stanza vuota, in profondità sottoterra. Sfortunatamente, Maj non può semplicemente andare a cercare la scatola. È molto buio nella cantina e andarci con una luce solleverebbe sospetti. Il suo unico modo per trovare il tesoro è controllare a distanza un robot aspirapolvere che si trova nella cantina.

La cantina è rappresentata come una griglia $H \times W$, dove le righe sono numerate da 0 a $H-1$ (dall'alto verso il basso) e le colonne sono numerate da 0 a $W-1$ (da sinistra a destra), il che significa che la cella in alto a sinistra è $(0,0)$ e la cella in basso a destra è $(H-1,W-1)$. La scatola con il tesoro è in una cella sconosciuta, eccetto la cella $(0,0)$. Ogni notte, il robot aspirapolvere si accende nell'angolo in alto a sinistra e si muove nella cantina.

Ogni notte, Maj può dare al robot una sequenza di istruzioni su come dovrebbe muoversi sotto forma di una stringa composta da caratteri "<", ">", "^" e "v". Formalmente, se il robot si trova sulla cella (r,c), non bloccata su nessun lato, "<" sposta il robot a sinistra nella cella $(r,c-1)$, ">" sposta il robot a destra nella cella $(r,c+1)$, "^" sposta il robot in alto nella cella $(r-1,c)$ e "v" sposta il robot in basso nella cella $(r+1,c)$.

Le pareti della cantina sono solide, quindi se il robot tenta di uscire dalla griglia, non succederà nulla. Anche la scatola è solida e non può essere spinta. Alla fine di ogni notte, il robot riferirà la sua posizione e tornerà nell'angolo in alto a sinistra.

Il tempo è essenziale, quindi Maj decide di trovare la scatola nel minor numero di notti possibile.

Interazione

Questo è un problema interattivo.

- Il tuo programma deve iniziare leggendo una riga con due numeri interi H e W: l'altezza e la larghezza della griglia.

- Quindi, il tuo programma dovrebbe interagire con il grader. In ogni round di interazione, devi stampare un punto interrogativo "`?`", seguito da una stringa non vuota s composta dai caratteri "`<`", "`>`", "`^`", "`v`". La lunghezza di questa stringa può essere al massimo di 20.000 caratteri. Quindi, il tuo programma deve leggere due numeri interi r, c ($0 \le r \le H-1$, $0 \le c \le W-1$), la posizione del robot dopo aver eseguito le istruzioni. Nota che il robot torna sempre a $(0, 0)$ dopo ogni query.
- Quando conosci la posizione della scatola, stampa "`!`" seguito dai due numeri interi r_b, c_b, la riga e la colonna della scatola ($0 \le r_b \le H-1$, $0 \le c_b \le W-1$). Infine, il tuo programma deve uscire senza fare ulteriori interrogazioni. Questo output finale non conta come query quando si determina il punteggio.

Assicurati di eseguire il `flush` sullo standard output dopo aver scritto una query, altrimenti il tuo programma potrebbe ricevere il giudizio "Time Limit Exceeded". In Python, `print()` esegue `flush` automaticamente. In C++, usa `cout << endl;` per scrivere una nuova riga e fare `flush`; se usi `printf`, usa `fflush(stdout)`.

Il grader non è adattivo, il che significa che la posizione della scatola viene determinata prima dell'inizio dell'interazione.

Assunzioni e Punteggio

- $1 \le H, W \le 50$.
- La scatola non si troverà mai a $(0, 0)$. Ciò significa che $H + W \ge 3$.
- Ogni query può contenere al massimo 20.000 istruzioni.
- Puoi inviare al massimo 2.500 query.

La tua soluzione verrà testata su una serie di casi di test. Se la tua soluzione fallisce su uno *qualsiasi* di questi casi di test (ad esempio segnalando la posizione errata della scatola (Wrong Answer), crash (Runtime Error), superamento del limite di tempo (Time Limit Exceeded), ecc.), riceverai 0 punti e il giudizio appropriato.

Se il tuo programma trova con successo la posizione della casella in *tutti* i casi di test, otterrai il verdetto AC e un punteggio calcolato come segue:

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

dove Q è il massimo numero di query usate su ogni test case. La stampa della risposta finale non conta come query. Il punteggio sarà arrotondato all'intero più vicino.

In particolare, per ricevere 100 punti, il tuo programma deve risolvere ogni test case utilizzando al massimo $Q = 2$ query. La tabella seguente mostra alcuni valori di Q e il punteggio associato.

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

Strumento di test

Per facilitare il testing della tua soluzione, forniamo un semplice strumento che puoi scaricare. Vedi "allegati" in fondo alla pagina del problema su Kattis. L'uso dello strumento è facoltativo e puoi modificarlo. Si noti che il programma di valutazione ufficiale su Kattis è diverso dallo strumento di test.

Esempio di utilizzo (con $H = 4$, $W = 5$ e la casella nascosta nella posizione $r = 2$, $c = 3$):

Per i programmi Python, per esempio `solution.py` (normalmente eseguito come `pypy3 solution.py`):

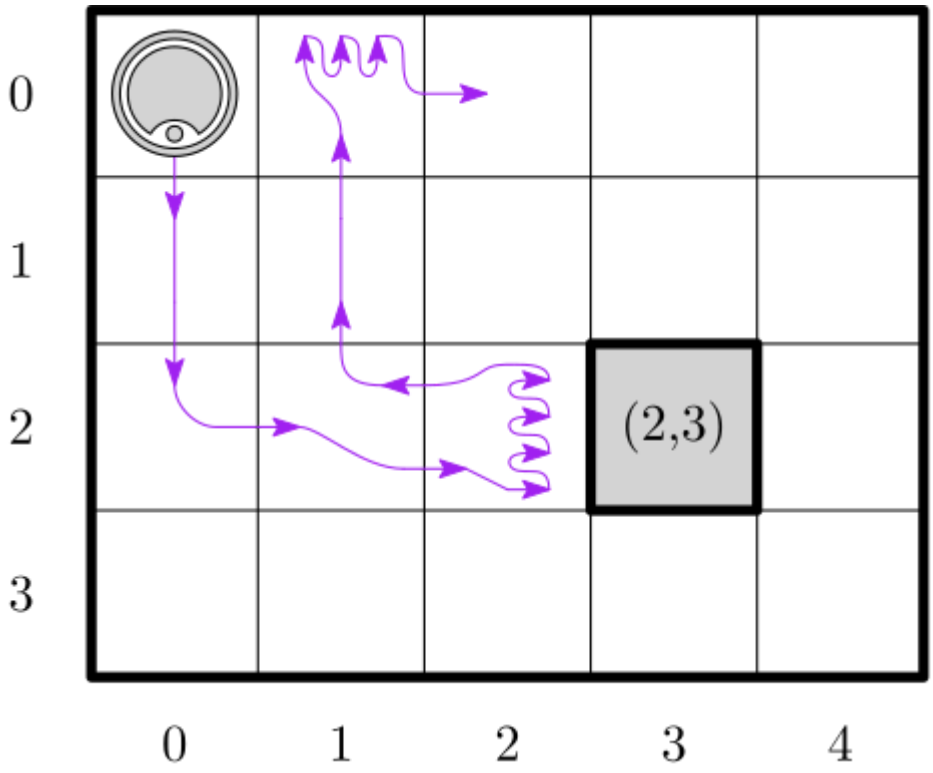
```
python3 testing_tool.py pypy3 solution.py <<< "4 5 2 3"
```

Per i programmi C++, prima compilalo (ad es. con `g++ -std=gnu++17 solution.cpp -o solution.out`), dopodichè esegui

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

Esempio

Considera il seguente caso di test di esempio. La griglia ha altezza $H = 4$ e larghezza $W = 5$, e la scatola si trova nella posizione $(r, c) = (2, 3)$. La figura seguente mostra il percorso del robot quando si seguono le istruzioni della prima query "`? vv>>>>>><^^^^>`", che risulta con il robot che finisce alla posizione $(r, c) = (0, 2)$. Prima della seconda query, il robot tornerà di nuovo nell'angolo in alto a sinistra $(0, 0)$. Quindi la soluzione emette un'altra query "`? >>>>>>>vvvvvvvvvv`" per cui il robot finisce nell'angolo in basso a destra $(r, c) = (3, 4)$. Ora la soluzione decide di indovinare la risposta, scrivendo "`! 2 3`", che è la posizione corretta della scatola.



output del grader	il tuo output
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3