Nájdi škatuľu

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

Maja pracuje v Lunde na univerzite. Robí výskum v oblasti robotiky. Nedávno sa dopočula, že kdesi v podzemí univerzity je ukrytá škatuľa s pokladom. Škatuľa stojí niekde v miestnosti, ktorá je inak prázdna. Maja potrebuje zistiť presnú polohu tejto škatule. Nemôže tam však ísť hľadať osobne - to by mohlo ľahko vzbudiť nežiadúcu pozornosť. A v podzemí je navyše úplná tma. Maja však prišla na geniálny plán: na prieskum použije robotický vysávač!

Miestnosť s pokladom má obdĺžnikový pôdorys. My si ju budeme reprezentovať ako mriežku s H riadkami a W stĺpcami. Riadky sú očíslované od 0 po $H-1$ zhora dole, stĺpce od 0 po $W-1$ zľava doprava. Každé políčko mriežky vieme teda popísať jeho súradnicami: (riadok, stĺpec). Škatuľa zaberá presne jedno políčko tejto mriežky, pričom je zaručené, že to nie je políčko $(0,0)$.

Každú noc môže Maja spraviť jeden experiment. Pri každom experimente robot začne na políčku $(0,0)$, teda v ľavom hornom rohu, následne sa bude pohybovať podľa postupnosti príkazov, ktorú mu Maja tú noc zadala, a potom nahlási výsledok.

Program, ktorý každú noc Maja zadá robotovi, je reťazec, v ktorom sa vyskytujú štyri znaky: "<", ">", "^" a "v". Ak sa robot nachádza na políčku (r,c), okolo ktorého je všade voľno, tak tieto príkazy majú nasledovný efekt: "<" pohne robota doľava na políčko $(r,c-1)$, ">" ho pohne doprava na políčko $(r,c+1)$, "^" ho pohne dohora na políčko $(r-1,c)$ a "v" pohne robota dodola na políčko $(r+1,c)$.

Steny miestnosti aj boky škatule sú pevné a robot nevie škatuľu tlačiť. Ak teda robot do niečoho narazí, namiesto vykonania príslušnej inštrukcie len ostane na mieste.

Na konci každej noci robot nahlási Maji súradnice, na ktorých skončil po vykonaní celého programu (a následne sa automaticky vráti domov za ňou, aby ho mohla preprogramovať).

Pomôžte Maji nájsť presné súradnice škatule za čo najmenej nocí.

Interaction

Ako už určite čakáš, táto úloha je interaktívna.

Tvoj program musí začať tým, že načíta riadok, v ktorom sú celé čísla H a W: rozmery mriežky.

Následne bude tvoj program interagovať s testovačom. V každom kole interakcie oznámiš testovaču program pre robota a testovač odsimuluje jeho pohyb a povie ti, kde skončil.

Formálne: Vždy, keď chce tvoj program poslať robota do miestnosti, musí vypísať jeden riadok a v ňom otáznik ("`?`"), medzeru a neprázdny reťazec s tvorený znakmi "`<`", "`>`", "`^`", "`v`". Každý reťazec s musí mať dĺžku nanajvýš $20\,000$. Následne musí tvoj program načítať zo vstupu jeden riadok, v ktorom budú dve celé čísla r, c ($0 \le r \le H-1,\ 0 \le c \le W-1$): súradnice políčka, kde robot skončil po vykonaní celého programu s. Pripomíname, že na začiatku vykonávania každého programu sa robot nachádza na políčku $(0, 0)$.

Keď tvoj program zistí, kde je škatuľa, oznámi to testovaču tak, že vypíše jeden riadok a v ňom výkričník ("`!`"), medzeru, r_b, medzeru a c_b, kde r_b, c_b ($0 \le r_b \le H-1,\ 0 \le c_b \le W-1$) sú súradnice políčka, kde stojí škatuľa. Následne musí tvoj program skončiť bez toho, aby čokoľvek ďalšie vypisoval.

Pri hodnotení tvojho riešenia výpis odpovede nepovažujeme za otázku.

Daj si pozor, aby si flushla výstup po každom riadku, ktorý vypíšeš. Ak tak neurobíš, môžeš dostať verdikt Time Limit Exceeded. V Pythone by malo stačiť používať `print()`, ktorý flushuje automaticky. V C++ `cout << endl;` po vypísaní znaku nového riadku aj flushne výstup. Ak používaš `printf`, po ňom sprav `fflush(stdout)`.

Testovač nie je adaptívny. Pri každom teste teda platí, že poloha škatule je vopred určená a nezávisí od toho, aké programy robotovi zadáš.

Constraints and Scoring

- $1 \le H, W \le 50$.
- Škatuľa nikdy nebude na súradniciach $(0, 0)$.
- Z toho vyplýva, že v každom vstupe bude $H + W \ge 3$.
- V každom programe pre robota môžeš mať nanajvýš $20\,000$ inštrukcií.
- Testovaču môžeš položiť nanajvýš $2\,500$ otázok. (Výpis odpovede nie je otázka.)

Tvoje riešenie bude otestované na veľa rôznych testoch. Ak čo len jeden z nich nevyrieši správne (napr. oznámi zlé súradnice škatule (Wrong Answer), skončí s chybou (Run Time Exception), prekročí časový limit (Time Limit Exceeded) a podobne), dostaneš za celú úlohu 0 bodov a verdikt zodpovedajúci typu problému, ktorý nastal.

Ak tvoj program *vždy* nájde správnu polohu škatule, dostaneš verdikt Accepted. Tvoje skóre za túto úlohu sa vypočíta nasledovným vzorcom:

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

kde Q je maximálny počet otázok, ktoré tvoj program potreboval na vyriešenie niektorého testu. (Pripomíname, že výpis odpovede sa nepočíta za otázku.) Tvoje skóre bude následne zaokrúhlené na najbližšie celé číslo.

Zo vzorca sa dá odvodiť, že na zisk 100 bodov je potrebné, aby tvoj program vyriešil každý test pomocou nanajvýš $Q = 2$ otázok. V nižšie uvedenej tabuľke máš niekoľko ďalších hodnôt Q a im zodpovedajúce počty bodov.

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

Testing Tool

Z Kattisu si môžeš stiahnuť testing tool k tejto úlohe. Nájdeš ho na spodku stránky pre túto úlohu (v sekcii Attachments).

Používanie tohto nástroja je dobrovoľné. Ak chceš, môžeš si ho ľubovoľne upraviť. (Pri skutočnom testovaní odovzdaného programu v Kattise sa používa iný testovač ako ten, čo ste dostali vy.)

Príklad použitia (pre rozmery $H = 4$, $W = 5$ a škatuľu na súradniciach $r = 2$, $c = 3$):

Ak programuješ v Pythone a tvoj program je v súbore `solution.py` (normálne by si ho spúšťala príkazom `pypy3 solution.py`), spusti:

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

Ak programuješ v C++, svoj program najskôr skompiluj

(napr. príkazom `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`)

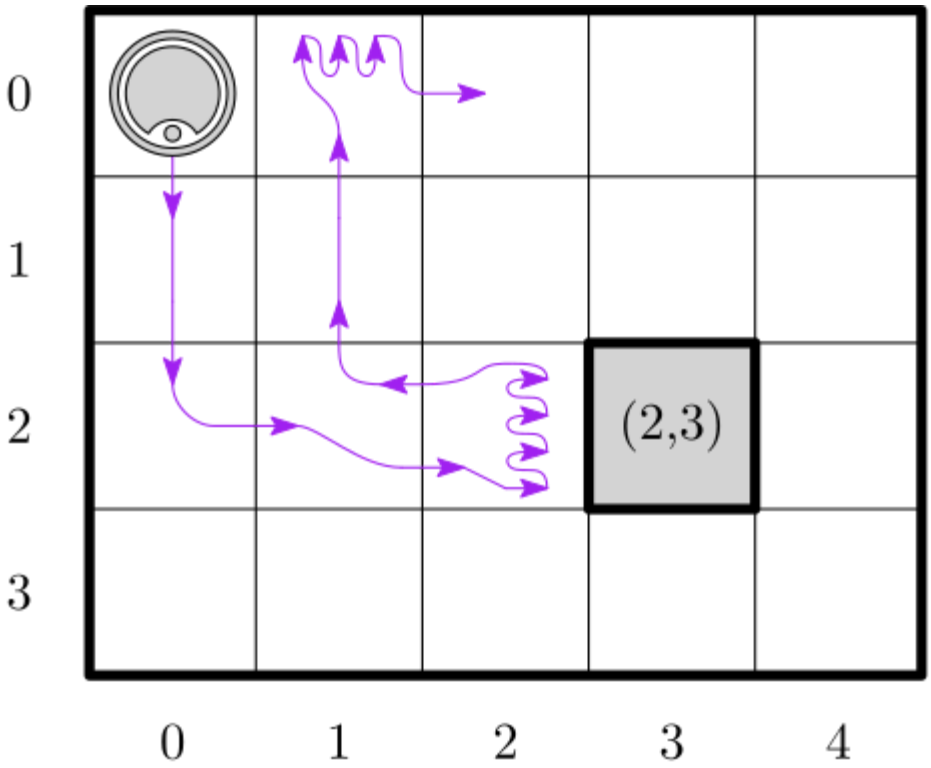
a potom spusti:

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

Example

V nižšie uvedenom príklade máme rozmery $H = 4$, $W = 5$ a škatuľu na súradniciach $r = 2$, $c = 3$.

Na obrázku je cesta robota, ktorý vykonáva prvú otázku -- teda "`? vv>>>>>>><^^^^^>`". Na konci tohto programu robot skončí na súradniciach $(r, c) = (0, 2)$.



Pred vykonaním druhej otázky sa robot vráti domov, druhú otázku (“`? >>>>>>>>vvvvvvvvvv`”) teda začne vykonávať znova na políčku $(0, 0)$. Vykonanie tohto druhého programu dostane robota do pravého dolného rohu, teda na políčko $(r, c) = (3, 4)$.

Následne sa ukážkové riešenie rozhodne tipnúť si odpoveď a zhodou okolností trafí tú správnu. Uhádnutú odpoveď vypíše ako riadok tvaru “`! 2 3`”.

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