Trouvez la boîte

Nom du problème	Find the Box
Limite de temps	1 seconde
Limite de mémoire	1 gigaoctet

Maj est une chercheuse en robotique qui travaille à l'université de Lund. Elle a appris qu'un trésor d'une valeur immense se trouve dans la cave de l'université. Ce trésor est dans une boîte qui se trouve dans une salle vide dans les profondeurs du sous-sol. Malheureusement, Maj ne peut pas simplement aller chercher la boîte elle-même. Il fait très noir dans la cave et y aller avec une lampe torche serait suspect. Le seul moyen à sa dispostion pour trouver le trésor est de contrôler à distance le robot-aspirateur qui demeure dans la cave.

La cave est représentée par une grille de taille $H \times W$, où les lignes sont numérotées de 0 à $H-1$ (haut en bas) et les colonnes sont numérotées de 0 à $W-1$ (gauche à droite), ce qui signifie que la case en haut à gauche a pour coordonnés $(0,0)$ et la case en bas à droite $(H-1, W-1)$. La boîte avec le trésor est dans un certaine case inconnue, différente de la case $(0,0)$. Chaque nuit, le robot-aspirateur commence dans la case en haut à gauche et se déplace dans la cave.

Chaque nuit, Maj peut donner au robot une suite d'instructions qui lui indique comment se déplacer, sous la forme d'une chaîne ne pouvant contenir qu'uniquement les caractères "<", ">", "^" et "v". Plus formellement, si le robot se trouve sur la case (r, c) qui n'est entourée par aucun obstacle, "<" déplace le robot d'une case vers la gauche sur la case $(r, c-1)$, ">" déplace le robot vers la droite sur la case $(r, c+1)$, "^" déplace le robot vers le haut sur la case $(r-1, c)$, et "v" déplace le robot vers le bas sur la case $(r+1, c)$.

Les murs de la cave sont solides, donc si le robot essaie de se déplacer en dehors de la grille il ne se passera rien (pas de mouvement du robot). La boîte est solide également, et ne peut pas être poussée. À la fin de chaque nuit le robot signale sa position, et retourne sur la case en haut à gauche.

Le temps est compté, donc Maj décide de trouver la boîte en aussi peu de nuits que possible.

Interaction

Ceci est un problème interactif.

- Votre programme doit commencer par lire une ligne avec deux entiers H et W : la hauteur et la largeur de la grille.
- Ensuite, votre programme doit interagir avec l'évaluateur (*grader*). À chaque tour d'interaction, vous devez afficher un point d'interrogation "`?`", suivi d'une chaîne de caractère non-vide s pouvant uniquement contenir les caractères "`<`", "`>`", "`^`", "`v`". La longueur de cette chaîne de caractères doit être au plus $20\,000$. Ensuite, votre programme doit lire deux entiers r, c ($0 \le r \le H-1,\ 0 \le c \le W-1$), la localisation du robot après avoir exécuté les instructions. Notez que le robot retourne toujours à la position $(0, 0)$ après chaque requête.
- Quand vous connaissez la localisation de la boîte, affichez "`!`" suivi de deux entiers r_b, c_b, la ligne et la colonne de la boîte ($0 \le r_b \le H-1,\ 0 \le c_b \le W-1$). Après cela, votre programme doit se terminer sans faire aucune requête supplémentaire. L'affichage final ne compte pas comme une requête lors du calcul de votre score.

Assurez-vous de synchroniser la sortie standard après avoir fait une requête, sinon votre programme risque d'être jugé comme Time Limit Exceeded. En Python, `print()` synchronise automatiquement. En C++, `cout << endl;` synchronise en plus d'afficher une nouvelle ligne ; si vous utilisez printf, alors utilisez `fflush(stdout)`.

L'évaluateur est non-adaptatif, ce qui signifie que la position de la boîte est déterminée avant le début de l'interaction.

Contraintes et score

- $1 \le H, W \le 50$.
- La boîte ne sera jamais sur la case $(0, 0)$. Cela signifie que $H + W \ge 3$.
- Chaque requête peut contenir au plus $20\,000$ instructions.
- Vous pouvez faire au plus $2\,500$ requêtes (afficher la réponse finale ne compte pas comme une requête).

Votre solution sera testée sur plusieurs tests. Si votre solution se trompe sur *n'importe lequel* de ces tests (par exemple en renvoyant la mauvaise position de la boîte (Wrong Answer), s'arrête brutalement ("crash") (Runtime Error), dépasse la limite de temps (Time Limit Exceeded), etc., vous recevrez 0 points et le verdict approprié.

Si votre programme trouve avec succès la position de la boîte pour *tous* les tests, vous recevrez le verdict Accepted, et un score calculé comme suit :

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

où Q est le *maximum* du nombre de requêtes utilisées parmi tous les tests. Afficher la réponse finale ne compte pas comme une requête. Le score sera arrondi à l'entier le plus proche.

En particulier, pour avoir 100 points, votre programme doit résoudre chaque test en utilisant au plus $Q = 2$ requêtes. Le tableau ci-dessous contient certaines valeurs de Q et le score associé.

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

Outil de test

Pour vous aider à tester votre solution, vous avez accès à un outil simple à télécharger. Vous le trouverez dans “attachments” en bas de la page Kattis du problème. L'utilisation de cet outil est optionnelle, et vous avez le droit de le modifier. Notez que l'évaluateur officiel de Kattis sera différent de cet outil de test.

Exemple d'utilisation (avec $H = 4$, $W = 5$, et la boîte cachée à la position $r = 2$, $c = 3$):

Pour un programme Python, par exemple nommé `solution.py` (normalement exécuté avec `pypy3 solution.py`):

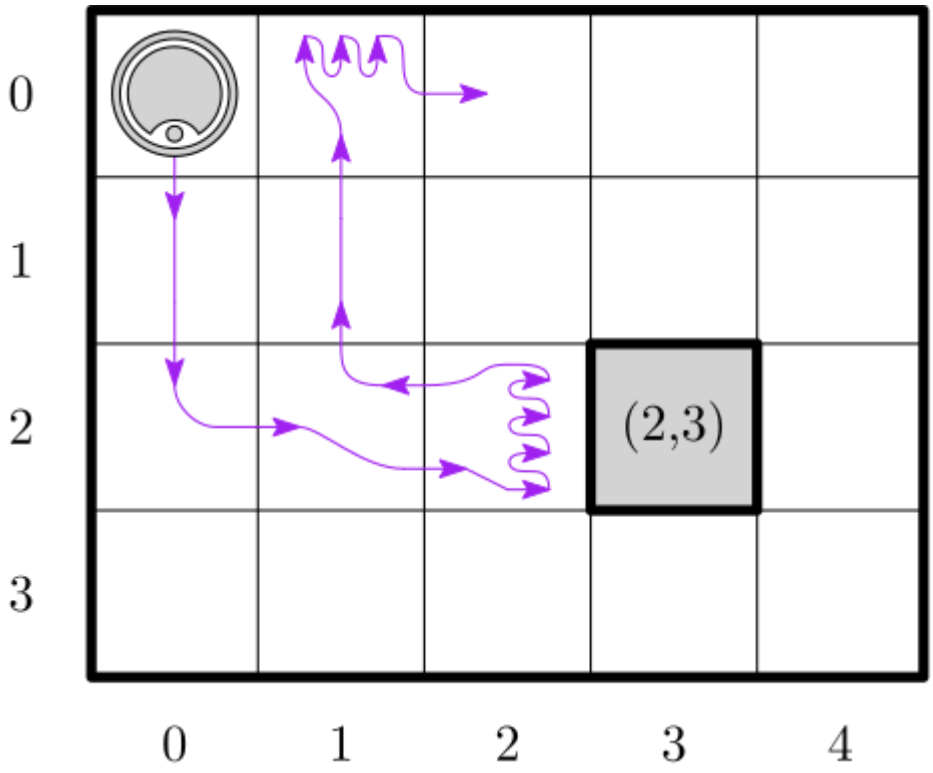
```
python3 testing_tool.py pypy3 solution.py <<<"4 5 2 3"
```

Pour un programme C++, par exemple nommé solution.cpp, compilez-le avec (e.g. with `g++ -g -O2 -std=gnu++17 -static solution.cpp -o solution.out`) et ensuite exécutez :

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

Exemple

Considérons le test d'exemple. La grille a une hauteur de $H = 4$ et une largeur de $W = 5$, et la boîte est à la position $(r, c) = (2, 3)$. La figure ci-dessous montre le chemin du robot lorsqu'il suit les instructions de la première requête “`? vv>>>>>><^^^^>`”, ce qui le fait terminer à la position $(r, c) = (0, 2)$. Avant la deuxième requête, le robot va retourner en haut à gauche en $(0, 0)$. Ensuite l'exemple de solution fait une deuxième requête “`? >>>>>>>>vvvvvvvvvv`” pour laquelle le robot va finir dans le coin en bas à droite $(r, c) = (3, 4)$. La solution décide ensuite de deviner la réponse en affichant “`! 2 3`”, ce qui est la bonne position de la boîte.



Sortie de l'évaluateur	Votre sortie
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3