Où est Charlie ?

Nom du problème	whereswaldo
Limit de temps	11 secondes
Limite de mémoire	1 gigaoctets

Il y a une certaine permutation $P = P_0, P_1, ..., P_{N-1}$ de longueur N, et il est garanti qu'elle est générée aléatoirement. La permutation contient les entiers $1, 2, 3, ..., N$ exactement une fois chacun, dans un ordre inconnu.

Vous pouvez choisir des positions l et r, et poser des questions de la forme : "Quelle est la valeur de $P_l + P_{l+1} + \cdots + P_r$?"

Votre but est de trouver la position de l'entier 1 dans P en posant aussi peu de questions que possible. Votre score dépendra du nombre de questions posées.

Interaction

Votre programme doit lire deux entiers sur une même ligne, T et N. T est le nombre tours sur lequelles votre prgramme sera testé, et N est la longueur de P.

Il y a ensuite T tours :

Quand un tour commence, vous pouvez démarrer les questions. Affichez une ligne avec "`? a b`" pour demander la somme des entiers entre les positions a et b inclus ($0 \leq a \leq b \leq N - 1$).

Après chaque question votre programme doit lire un entier, la somme des valeurs dans cet intervalle.

Une fois que vous avez trouvé la position du 1, affichez une ligne de la forme "`! i`", où i est la position telle que $P_i = 1$. Après avoir affiché cette ligne, le tour suivant commence.

Assurez-vous de synchroniser la sortie standard après avoir posé une question, sinon votre programme risque de recevoir le résultat 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)`.

Contraintes et score

Votre programme sera testé sur **un unique test, avec** $N = T = 1000$. La permutation dans chaque test est garantie d'être **générée aléatoirement**.

Si votre solution donne un résultat faux sur n'importe lequel des tours, votre soumission sera jugée comme *Wrong Answer*.

Sinon, votre score sera calculé de la manière suivante :

$$\text{score} = \min\left(220 - \frac{M}{2500}, 100\right) \text{ points},$$

où M est le nombre de questions que votre programme a posé au total sur les T tours.

Votre score sera arrondi à l'entier le plus proche. Si le score est négatif, il vaudra 0 points.

Ainsi, si vous posez plus de $550\,000$ questions alors vous recevrez 0 points, et si vous posez $300\,000$ ou moins vous recevrez 100 points. Entre ces bornes, vos scores croissent de manière linéaire.

Outil de test

Pour vous aider à tester vos solutions, vous avez accès à un outil que vous pouvez 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 d'en modifier le code. Notez le programme officiel de test (*grader*) de Kattis est différent de cet outil qui vous est fourni.

Exemple d'utilisation (avec T=1000, N=10):

Pour les programmes Python, par exemple appelé `solution.py` (qui peut être exécutée avec `pypy3 solution.py`):

```
python3 testing_tool.py pypy3 solution.py <<<"1000 10"
```

Pour les programmes C++, compilez d'abord avec (e.g. with `g++ -std=gnu++17 solution.cpp -o solution.out`) et ensuite exécutez :

```
python3 testing_tool.py ./solution.out <<<"1000 10"
```

Exemple

Dans le test d'exemple, $T = 2$ et $N = 10$. Pour le premier de ces deux tours, disons que la permutation cachée est “`6 10 8 7 9 1 2 4 5 3`”. La première question `? 0 9` demande la

somme de tous ces nombres, qui est effectivement 55, et la deuxième question `? 0 4` demande $6 + 10 + 8 + 7 + 9 = 40$.

Sortie du grader	Votre sortie
2 10	
	? 0 9
55	
	? 0 4
40	
	? 5 5
1	
	! 5
	? 0 0
1	
	! 0