¿Dónde está Wally?

Nombre del problema	whereswaldo
Límite de tiempo	11 segundos
Límite de memoria	1 gigabyte

Hay una permutación oculta $P_0, P_1, ..., P_{N-1}$ de longitud N, y se garantiza que esta ha sido generada de manera uniformemente aleatoria. La permutación contiene los números $1, 2, 3, ..., N$ una única vez cada uno, en un cierto orden desconocido.

Puedes elegir dos posiciones l y r, para hacer preguntas de la forma: "¿Cuál es la suma de $P_l + P_{l+1} + \cdots + P_r$?"

Tu objetivo es encontrar la posición del número 1 en P usando el mínimo número de preguntas posibles. Se te puntuará dependiendo del número de preguntas usadas.

Interacción

En primer lugar, tu programa debe leer dos enteros en la misma línea, T y N.

T es el número de rondas en las que tu programa será puesto a prueba, y N es la longitud de P.

Después de esto hay T rondas:

Cuando una ronda comienza puedes empezar a hacer preguntas. Imprime una línea con "`? a b`" para preguntar la suma de los números entre las posiciones a y b incluyendo ambos extremos ($0 \leq a \leq b \leq N - 1$).

Después de cada pregunta tu programa debe leer un entero, la suma de los números en el intervalo.

Una vez que hayas encontrado la posición de 1, imprime una linea de la forma "`! i`", donde i es el índice tal que $P_i = 1$. Después de imprimir esto, la siguiente ronda empezará.

Asegúrate de hacer flush después de hacer una pregunta, o tu programa podría recibir el veredicto *Time Limit Exceeded*.

En Python, `print()` hace flush automáticamente. En C++, `cout << endl;` también hace flush además de añadir un salto de línea; si usas printf, usa `fflush(stdout)`.

Restricciones y Puntuación

Tu programa será puesto a prueba en **un solo testcase, con** $N = T = 1000$. Se garantiza que la permutación en cada ronda está **generada aleatoriamente**.

Si tu solución da una respuesta errónea en cualquiera de las rondas, tu envío recibirá el veredicto *Wrong Answer*.

En caso contrario, la puntuación se calcula de la siguiente manera:

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

donde M es el número de preguntas que tu programa ha usado en total en las T rondas.

La puntuación se redondeará al entero más cercano. Si la puntuación se hace negativa contará como cero puntos.

Por tanto, si usas más de $550\,000$ preguntas recibirás 0 puntos, y si usas $300\,000$ o menos preguntas recibirás 100 puntos. Entre estos valores, tu puntuación aumenta linealmente.

Herramienta de testing

Para facilitarte el testing de tu solución, tienes una sencilla herramienta que puedes descargar. Ver “attachments” al final de la página del problema en kattis. El uso de la herramienta es opcional, y se te permite modificarla. Ten en cuenta que el programa oficial con el que calcula las puntuaciones kattis no es el mismo que esta herramienta.

Ejemplo de uso (con T=1000, N=10):

Para programas en Python, digamos `solution.py` (habitualmente ejecutado como `pypy3 solution.py`):

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

Para programas en C++, primero compilamos (por ejemplo con `g++ -std=gnu++17 solution.cpp -o solution.out`) y luego ejecutamos:

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

Testcase de ejemplo

En el testcase de ejemplo, $T = 2$ y $N = 10$. En la primera de estas dos rondas, digamos que la permutación oculta es "`6 10 8 7 9 1 2 4 5 3`". La primera pregunta `? 0 9` pregunta la suma de todos los números, que es 55, y la segunda pregunta `? 0 4` pregunta por $6 + 10 + 8 + 7 + 9 = 40$.

grader output	tu output
2 10	
	? 0 9
55	
	? 0 4
40	
	? 5 5
1	
	! 5
	? 0 0
1	
	! 0