Balanza

Amina tiene seis monedas, numeradas de 1 a 6. Sabe que los pesos de las monedas son todos distintos entre sí. Desea ordenarlas de acuerdo a su peso. Para este propósito, ha desarrollado un nuevo tipo de balanza.

Una balanza tradicional consta de dos platos. Para utilizarla, se coloca una moneda en cada plato y la balanza determina cuál de las monedas es la más pesada.

La nueva balanza de Amina es más compleja. Tiene cuatro platos, rotulados A, B, C, y D. La balanza tiene cuatro configuraciones posibles, cada una de las cuales responde una pregunta diferente acerca de las monedas. Para utilizar la balanza, Amina debe colocar exactamente una moneda en cada uno de los platos A, B, y C. Además, cuando se utiliza la cuarta configuración debe colocar exactamente una moneda en el plato D.

Las cuatro configuraciones hacen que la balanza responda las siguientes preguntas:

1. ¿Cuál de las monedas en los platos A, B, y C es la más pesada?
2. ¿Cuál de las monedas en los platos A, B, y C es la más liviana?
3. ¿Cuál de las monedas en los platos A, B, y C es la mediana? (Es decir, la moneda que no es ni la más pesada ni la más liviana de las tres.)
4. De las monedas en los platos A, B, y C, considérese únicamente aquellas que son más pesadas que la moneda en el plato D. Si existe al menos una moneda en tales condiciones, ¿cuál de ellas es la más liviana? De lo contrario, si no existe ninguna moneda en tales condiciones, ¿cuál de las monedas en los platos A, B, y C es la más liviana?

Tarea

Escriba un programa que ordene las seis monedas de Amina de acuerdo a su peso. El programa puede consultar la balanza de Amina para comparar los pesos de las monedas. El programa recibirá varios casos de prueba para resolver, cada uno de los cuales corresponderá a un nuevo conjunto de seis monedas.

El programa deberá implementar las funciones `init` y `orderCoins`. En cada corrida del programa, el evaluador llamará primero a `init` exactamente una vez. En esta llamada el programa recibe la cantidad de casos de prueba, y le permite inicializar variables. A continuación, el evaluador llamará a `orderCoins()` exactamente una vez por caso de prueba.

- `init(T)`
  - `T`: La cantidad de casos de prueba que el programa deberá resolver en esta corrida. `T` es un entero en el rango $1, \ldots, 18$.
  - Esta función no retorna ningún valor.

- `orderCoins()`
  - Esta función es llamada exactamente una vez por caso de prueba.
  - La función debe determinar el orden correcto de las monedas de Amina, llamando a las funciones `getLightest()`, `getHeaviest()`, `getMedian()`, y/o `getNextLightest()`.
  - Una vez que la función determina el orden correcto, deberá reportarlo llamando a la función `answer()`.
  - Luego de llamar a `answer()`, la función `orderCoins()` deberá retornar. No retorna ningún valor.

Se puede utilizar las siguientes funciones del evaluador en el programa:

- `answer(W)` — el programa deberá utilizar esta función para reportar la respuesta hallada.
  - `W`: Un arreglo de longitud 6 con el orden correcto de las monedas. `W[0]` a `W[5]` deberán contener los números de moneda (es decir, números de 1 a 6) en orden desde la moneda más liviana hasta la más pesada.
  - El programa únicamente deberá llamar a esta función desde `orderCoins()`, exactamente una vez por caso de prueba.
  - Esta función no retorna ningún valor.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — estas funciones se corresponden con las configuraciones 1, 2 y 3 de la balanza de Amina, respectivamente.
  - `A`, `B`, `C`: Las monedas que se colocan en los platos A, B, y C, respectivamente. `A`, `B`, y `C` deberán ser tres enteros distintos, entre 1 y 6 inclusive.
  - Cada una de las funciones retorna uno de los números `A`, `B`, y `C`: el número que corresponda a la moneda apropiada. Por ejemplo, `getHeaviest(A, B, C)` retorna el número que corresponde a la más pesada de las tres monedas dadas.
- `getNextLightest(A, B, C, D)` — esta función corresponde a la configuración 4 de la balanza de Amina
  - `A`, `B`, `C`, `D`: Las monedas que se colocan en los platos A, B, C, y D, respectivamente. `A`, `B`, `C`, y `D` deberán ser cuatro enteros distintos, cada uno entre 1 y 6 inclusive.
  - La función retorna uno de los números `A`, `B`, y `C`: el número de la moneda seleccionada por la balanza, de acuerdo a lo descrito anteriormente para la configuración 4. Es decir, la moneda retornada es la más liviana entre aquellas monedas de los platos A, B, y C que son mas pesadas que la moneda en el plato D; o, si ninguna de ellas es más pesada que la moneda en el plato D, la moneda retornada es simplemente la más liviana de las tres monedas en los platos A, B, y C.

Puntaje

Este problema no tiene subtareas. En cambio, el puntaje se basa en la cantidad total de pesadas (es decir, la cantidad total de llamadas a las funciones `getLightest()`, `getHeaviest()`,

`getMedian()` y/o `getNextLightest()` del evaluador) que realiza el programa.

El programa será ejecutado múltiples veces, con múltiples casos de prueba en cada corrida. Sea r el número de corridas del programa. Este número está fijado por los datos de test utilizados en la evaluación. Si el programa no produce el orden correcto de las monedas en algún caso de prueba de alguna corrida, obtendrá automáticamente 0 puntos. De lo contrario, el puntaje de cada corrida se calcula de manera independiente como se indica a continuación.

Sea Q el menor número tal que siempre es posible ordenar una secuencia de seis monedas utilizando a lo sumo Q pesadas en la balanza de Amina. Para hacer la tarea más desafiante, el valor de Q no será revelado en este enunciado.

Sea $Q + y$ la máxima cantidad de pesadas que realiza el programa, de entre todos los casos de prueba de todas las corridas, con y entero no negativo. Luego, considérese una corrida particular del programa. Sea $Q + x$ la máxima cantidad de pesadas que realiza el programa, de entre todos los T casos de prueba de esta corrida, con x entero no negativo. (Si el programa utiliza menos de Q pesadas en todos los casos de prueba, entonces $x = 0$.) El puntaje para esta corrida será $\frac{100}{r((x+y)/5+1)}$, truncado a dos dígitos después del punto decimal.

En particular, si el programa realiza a lo sumo Q pesadas en cada uno de los casos de prueba de cada una de las corridas, obtendrá 100 puntos.

Ejemplo

Supongamos que las monedas se ordenan $3\ 4\ 6\ 2\ 1\ 5$ desde la más liviana a la más pesada.

Llamada a función	Valores retornados	Explicación
getMedian(4, 5, 6)	6	La moneda número **6** es la mediana entre las monedas número **4**, **5**, y **6**.
getHeaviest(3, 1, 2)	1	La moneda número **1** es la más pesada entre las monedas número **1**, **2**, y **3**.
getNextLightest(2, 3, 4, 5)	3	Las monedas número **2**, **3**, y **4** son todas más livianas que la moneda número **5**, por lo cual la mas liviana de ellas (**3**) es retornada.
getNextLightest(1, 6, 3, 4)	6	Las monedas número **1** y **6** son ambas más pesadas que la número **4**. Entre las monedas **1** y **6**, la **6** es la más liviana.
getHeaviest(3, 5, 6)	5	La moneda **5** es la más pesada entre **3**, **5** y **6**.
getMedian(1, 5, 6)	1	La moneda **1** es la mediana entre las monedas **1**, **5** y **6**.
getMedian(2, 4, 6)	6	La moneda **6** es la mediana entre las monedas **2**, **4** y **6**.
answer([3, 4, 6, 2, 1, 5])		El programa halló la respuesta correcta para este caso de prueba.

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

- línea 1: T —- la cantidad de casos de prueba

- cada una de las líneas 2 a $T+1$: una secuencia de 6 números distintos entre 1 y 6: el orden de las monedas de la más liviana a la más pesada.

Por ejemplo, una entrada que consiste de dos casos de prueba donde las monedas se ordenan **1 2 3 4 5 6** y **3 4 6 2 1 5** se ve de la siguiente manera:

```
2
1 2 3 4 5 6
3 4 6 2 1 5
```

El evaluador de ejemplo imprime el arreglo que se pasa como parámetro a la función `answer()`.