Scales

Amina tiene seis monedas, numeradas de 1 a 6. Ella sabe que todas las monedas tienen diferentes pesos. A ella le gustaría ordenarlas acorde a su peso. Para este propósito ella ha desarrollado un nuevo tipo de balanza.

Una balanza tradicional tiene dos platos. Para usar dicha balanza, tu colocas una moneda en cada plato y la balanza determinará que moneda es más pesada.

La nueva balanza de Amina es más compleja. Esta tiene cuatro platos, etiquetados A, B, C y D. La balanza tiene cuatro configuraciones diferentes, cada una de las cuales responde una pregunta diferente acerca de las monedas. Para usar la balanza, Amina debe colocar exáctamente una moneda en cada uno de los platos A, B, y C. Adicionalmente, en la cuarta configuración ella también debe colocar exáctamente una moneda en el plato D.

Las cuatro configuraciones indican a la balanza responder las siguientes cuatro 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 ligera?
3. ¿Cuál de las monedas en los platos A, B y C es la mediana? (Esta es la moneda que no es ni la más pesada ni la más ligera)
4. Entre las monedas en los platos A, B y C, considera solo las monedas que son más pesadas que la moneda en el plato D. Si hay tales monedas, ¿cuál de estas es la más ligera?. En caso contrario, si no hay tales monedas, ¿cuál de las monedas en los platos A, B y C es la más ligera?

Task

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. A su programa le serán dados varios casos de prueba para resolver, cada uno correspondiendo a un nuevo conjunto de seis monedas.

Su programa debe implementar las funciones `init` y `orderCoins`. Durante cada corrida de tu programa, el grader primero llamará a `init` exáctamente una vez. Esto le dará el número de casos de prueba y le permitirá inicializar las variables que considere necesarias. Entonces, el grader llamará a `orderCoins()` una vez por caso de prueba.

- `init(T)`
  - `T`: El número de casos de prueba que tu programa debeŕa resolver durante esta corrida. `T` es un entero del rango $1, \ldots, 18$.
  - Esta función no tiene valor de retorno.
- `orderCoins()`

- Esta función es llamada exáctamente una vez por caso de prueba.
- Esta función debe determinar el orden correcto de las monedas de Amina llamando a las funciones del grader `getHeaviest()`, `getLightest()`, `getMedian()`, y/o `getNextLightest()`.
- Una vez que la función sabe el orden correcto, debe reportarlo llamando a la función del comparador `answer()`.
- Luego de llamar a `answer()`, su implementación de la función `orderCoins()` debe terminar. No tiene valor de retorno.

Puedes usar las siguientes funciones del grader en tu programa:

- `answer(W)` — su programa deberá utilizar esta función para reportar la respuesta que se consiguió.
  - `W`: Un arreglo de longitud 6 que contiene el orden correcto de las monedas. `C[0]` hasta `C[5]` que deberán ser los números de las monedas (i.e., los números del 1 al 6) desde la moneda más ligera a las más pesada.
  - Tu programa solo debe llamar esta función desde `orderCoins()`, una vez por caso de prueba.
  - Esta función no tiene valor de retorno.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — Estas funciones corresponden a las configuraciones 1, 2 y 3 respectivamente para la balanza de Amina.
  - `A`, `B`, `C`: Las monedas son puestas en los platos A, B, y C, respectivamente. `A`, `B`, y `C` deben ser tres enteros diferentes, cada uno entre 1 y 6 inclusive.
  - Cada función retorna uno de los números `A`, `B`, y `C`: el número de la moneda apropiada. Por ejemplo, `getHeaviest(A, B, C)` retorna el número de la moneda más pesada de las tres monedas dadas.
- `getNextLightest(A, B, C, D)` — esta corresponde a la configuración 4 para la balanza de Amina
  - `A`, `B`, `C`, `D`: Las monedas son puestas en las balanzas A, B, C, y D, respectivamente. `A`, `B`, `C`, y `D` deben ser cuatro enteros diferentes, cada una entre 1 y 6 inclusive.
  - La función retorna uno de los números `A`, `B`, y `C`: el número de la moneda seleccionada por la escala como fue indicado anteriormente para la configuración 4. Esto es, la moneda retornada es la más ligera entre las monedas en los platos A, B, y C que son más pesadas que la moneda en el plato D; ó, si ninguna de ellas es más pesada que la moneda en el plato D, la moneda retornada es símplemente la más ligera de las tres monedas en los platos A, B, and C.

Scoring

No hay subtareas en este problema. En cambio, su puntaje se basará en cuantas veces su programa utiliza la balanza (número total de llamadas a las funciones del grader `getLightest()`,

`getHeaviest()`, `getMedian()` y/o `getNextLightest()`).

Su programa será corrido varias veces con múltiples casos de prueba en cada corrida. Siendo r el número de corridas de su programa. Este número es fijado por los datos de prueba. Si su programa no ordena de las monedas correctamente en ninguna corrida, obtendrá 0 puntos. De otra manera, cada corrida es puntuada individualmente como sigue.

Sea Q el número más pequeño tal que es posible ordenar cualquier secuencia de seis monedas usando Q veces la balanza de Amina. Para hacer esta tarea más retadora, no revelamos el valor de Q aquí.

Suponga que el mayor número de usos de la balanza en todos los casos de prueba de todas las corridas es $Q + y$ para algún entero y. Luego, considere una sola corrida de su programa. Sea el mayor número de usos de la balanza entre todos los T casos de prueba en esta corrida $Q + x$ para algún entero no negativo x. (Si usted utiliza menos de Q veces la balanza para cada caso de prueba, entonces $x = 0$.) Entonces, el puntaje para esta corrida será $\frac{100}{r((x+y)/5+1)}$, redondeando *hacia abajo* a dos dígitos después del punto decimal.

En particular, si tu programa hace a lo sumo Q usos de la balanza en cada corrida, obtendrás 100 puntos.

Example

Suponga que las monedas están ordenadas $3\ 4\ 6\ 2\ 1\ 5$ de la más ligera a la más pesada.

LLamada a la función	Retorna	Explicación
getMedian(4, 5, 6)	6	La moneda 6 es la mediana entre las monedas 4, 5, y 6.
getHeaviest(3, 1, 2)	1	La moneda 1 es la más pesada entre las monedas 1, 2, y 3.
getNextLightest(2, 3, 4, 5)	3	Las monedas 2, 3, y 4 son todas más ligeras que la moneda 5, entonces la más ligera de ellas (3) es retornada.
getNextLightest(1, 6, 3, 4)	6	Las monedas 1 y 6 ambas son más pesadas que la moneda 4. Entre las monedas 1 y 6, la moneda 6 es la más ligera.
getHeaviest(3, 5, 6)	5	La moneda 5 es la más ligera entre 3, 5 y 6.
getMedian(1, 5, 6)	1	La moneda 1 es la mediana entre 1, 5 y 6.
getMedian(2, 4, 6)	6	La moneda 6 es la mediana entre 2, 4 y 6.
answer([3, 4, 6, 2, 1, 5])		El programa encontró la solución para este caso de prueba.

Sample grader

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

- línea 1: T —- el número de casos de prueba
- cada una de las líneas desde 2 a $T + 1$: una secuencia de 6 números distintos desde 1 a 6: el orden de las monedas desde la más ligera a la más pesada.

Por ejemplo, una entrada consiste de dos casos de prueba donde las monedas son ordenadas **1 2 3 4 5 6** y **3 4 6 2 1 5** se ve como sigue:

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

El comparador de ejemplo imprime el arreglo que fué pasado como parámetro a la función `answer()`.