Balanzas

Amina posee seis monedas, enumeradas del 1 al 6. Ella sabe que todas las monedas poseen un peso distinto. Amina desea ordenarlas de acuerdo con su peso. Para este proposito, ha desarrollado un nuevo tipo de balanza.

Una balanza tradicional posee dos bandejas. Para usar este tipo de balanza, debes colocar una modena en cada bandeja y la balanza determinará cuál moneda es más pesada.

El nuevo sistema de balanzas de Amina es más complejo. Este tiene cuatro bandejas, etiquetadas A, B, C, y D. La balanza posee cuatro configuraciones distintas, cada una de las cuales responde a una pregunta distinta sobre las monedas. Para usar la balanza, Anima debe colocar exactamente una moneda en cada una de las bandejas A, B, y C. Adicionalmente, deberá colocar exactamente una moneda en la bandeja D.

Las cuatro configuraciones instruirán a la balanza a que responda las siguientes cuatro preguntas:

1. ¿Cuál de las monedas en las bandejas A, B, y C es la más pesada?
2. ¿Cuál de las monedas en las bandejas A, B, y C es la más liviana?
3. ¿Cuál de las monedas en las bandejas A, B, y C es la mediana? (Esta es la moneda que no es ni la más pesada ni la más liviana de las tres.)
4. Dentro de las monedas en las bandejas A, B, y C, considerar sólo las monedas que son más pesadas que la moneda en la bandeja D. Si existen tales monedas, ¿Cuál de las monedas es la más liviana? De lo contrario, si no hay ninguna moneda como tal, ¿Cuál de las monedas en las bandejas A, B, y C es la más liviana?

Tarea

Escribe un programa que ordene las seis monedas de Amina de acuerdo con su peso. El programa puede indicarle a la balanza de Amina que compare los pesos de las monedas. A tu programa le serán dados varios casos de prueba a resolver, cada uno correspondiente con un nuevo conjunto de seis monedas.

Tu programa debe implementar las funciónes `init` y `orderCoins`. Durante cada corrida, el grader primero llamará a `init` exactamente una vez. Esto te dará el número de casos de prueba y te permitirá inicializar cualquier variable. Luego, el grader llamará `orderCoins` una vez por cada caso de prueba.

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

- `orderCoins()`
  - Esta función es llamada exactamente una vez por cada caso de prueba.
  - Esta función debe determinar el orden correcto de las monedas de Amina al llamar las funciónes del grader `getLightest()`, `getHeaviest()`, `getMedian()`, y/o `getNextLightest()`.
  - Una vez que la función conozca el orden correcto, esta deberá reportarlo llamando la función del grader `answer()`.
  - Despues de llamar `answer()`, la función `orderCoins()` debe retornar. Esta no posee valor de retorno.

Puedes usar las siguientes funciones del grader en tu programa:

- `answer(W)` — tu programa debe usar esta función para reportar la respuesta que ha encontrado.
  - `W`: Un arreglo de longitud 6 conteniendo el orden correcto de las monedas. `W[0]` hasta `W[5]` debe contener los valores de las monedas (números de 1 al 6) en el orden de la moneda más liviana a la más pesada.
  - Tu programa sólo deberá llamar esta función desde `orderCoins()`, una vez por cada caso de prueba.
  - Esta función no posee valor de retorno.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — estas corresponden con las cofiguraciones 1, 2 y 3 para la balanza de Amina.
  - `A`, `B`, `C`: Las monedas que son colocadas en las bandejas A, B, y C, respectivamente. `A`, `B`, y `C` deben ser tres enteros distintos, cada uno dentro del rango 1 y 6 inclusive.
  - Cada función retorna uno de los números `A`, `B`, y `C`: el valor de la moneda apropiada. Por ejemplo, `getHeaviest(A, B, C)` retorna el valor más pesado de tres monedas dadas.
- `getNextLightest(A, B, C, D)` — esta corresponde a la configuración 4 de la balanza de Amina.
  - `A`, `B`, `C`, `D`: Las monedas que son colocadas en las bandejas A, B, C, y D, respectivamente. `A`, `B`, `C`, y `D` deben ser cuatro enteros distintos, cada uno dentro del rango 1 al 6 inclusive.
  - La función retorna uno de los números `A`, `B`, y `C`: el número de la moneda seleccionada por la balanza según fue descrito anteriormente para la configuración 4. Esto es, la moneda retornada es la más liviana dentro de las monedas colocadas en las bandejas A, B, y C que son más pesadas que la moneda en la bandeja D; o, si ninguna de ellas es más pesada que la moneda en la bandeja D, la moneda retornada es simplemente la más liviana de las tres monedas en las bandejas A, B, y C.

Puntuación

No hay sub-tareas en este problema. En su lugar, tu puntuación se basará en cuántas pesadas (número total de llamadas a la funciones del grader `getLightest()`, `getHeaviest()`,

`getMedian()` y/o `getNextLightest()`) hace tu programa.

Tu programa será corrido múltiples veces con múltiples casos de prueba en cada ejecución. Sea r el número de ejecuciones de tu programa. Este número será asignado por la data de prueba. Si tu programa no ordena las monedas correctamente en ninguno de los casos de cualquier corrida, obtendrá 0 puntos. De lo contrario, las ejecuciones serán calificadas individualmente de acuerdo a lo siguiente:

Sea Q el número más pequeño tal que sea posible ordenar cualquier secuencia de seis monedas usando Q pesadas en la balanza de Amina. Para hacer la tarea más retadora, no revelaremos aquí el valor de Q.

Supón que el mayor número de pesadas dentro de todos los casos de todas las ejecuciones es $Q + y$ para algún entero y. Entonces, considera una única ejecución de tu programa. Sea Q +x el número mas grande de pesadas dentro de todos los T casos en esta ejecución para algú entero no-negativo x. (Si usas menos de Q pesadas para cada caso de prueba, entonces $x = 0$.) Entonces, la puntuación para esta ejecución será $\frac{100}{r((x+y)/5+1)}$, redondeada hacia *abajo* a dos digitos después del punto decimal.

En particular, si tu programa hace a lo sumo Q pesadas en cada caso de prueba de cada ejecución, obtendrás 100 puntos.

Ejemplo

Supón que la monedas están ordenadas $3\ 4\ 6\ 2\ 1\ 5$ desde la más liviana hasta las mas pesada.

Funcion de llamada	Retorna	Explicacion
getMedian(4, 5, 6)	6	Moneda 6 es la mediana dentro de las monedas 4, 5, y 6.
getHeaviest(3, 1, 2)	1	Moneda 1 es la más pesada dentro de las monedas 1, 2, y 3.
getNextLightest(2, 3, 4, 5)	3	Moneda 2, 3, y 4 son todas mas livianas que las monedas 5, de manera que la más liviana dentro de ellas (3) es retornada.
getNextLightest(1, 6, 3, 4)	6	Monedas 1 y 6 son ambas más pesadas que la moneda 4. Entre las monedas 1 y 6, la moneda 6 es la más liviana.
getHeaviest(3, 5, 6)	5	Moneda 5 es la más pesada dentro de las monedas 3, 5 y 6.
getMedian(1, 5, 6)	1	Moneda 1 es la mediana dentro de las monedas 1, 5 y 6.
getMedian(2, 4, 6)	6	Moneda 6 es la mediana dentro de las monedas 2, 4 y 6.
answer([3, 4, 6, 2, 1, 5])		El prorama encontro la respuesta correcta para este caso de prueba.

Grader de Ejemplo

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

- linea 1: T —- el número de casos de prueba.
- cada una de las lineas desde 2 to $T + 1$: una secuencia de 6 números distintos desde 1 to 6: el

orden de las monedas desde la más liviana a la más pesada.

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

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

El grader de ejemplo imprimirá el arreglo que fue pasado como parametro a la función `answer()`.