Calabozos

Robert está diseñando un nuevo juego de computadora. El juego involucra un héroe, n oponentes y $n+1$ calabozos. Los oponentes son numerados de 0 a $n-1$ y los calabozos son numerados de 0 a n. El oponente i ($0 \leq i \leq n-1$) está ubicado en el calabozo i y tiene fuerza $s[i]$. No hay oponente en el calabozo n.

El héroe inicia entrando en el calabozo x, con fuerza z. Cada vez que el héroe entra en algún calabozo i ($0 \leq i \leq n-1$), se enfrenta al oponente i, y ocurre una de las siguientes situaciones:

- Si la fuerza del héroe es mayor o igual a la fuerza del oponente $s[i]$, el héroe gana. Esto causa que la fuerza del héroe **aumente** en $s[i]$ ($s[i] \geq 1$). En este caso el héroe entra al calabozo $w[i]$ siguiente ($w[i] > i$).
- De lo contrario, el héroe pierde. Esto causa que la fuerza del héroe **aumente** en $p[i]$ $(p[i] \geq 1)$. En este caso el héroe entra en el calabozo $l[i]$ siguiente.

Nota que $p[i]$ puede ser menor, igual o mayor que $s[i]$. También, $l[i]$ puede ser menor, igual, o mayor que i. Sin importar el resultado del enfrentamiento, el oponente permanece en el calabozo i y mantiene fuerza $s[i]$.

El juego termina cuando el héroe entra en el calabozo n. Se puede observar que el juego termina después de un número finito de enfrentamientos, sin importar el calabozo y la fuerza inicial.

Robert te pide que pruebes su juego ejecutando q simulaciones. Por cada simulación, Robert define un calabozo inicial x y una fuerza inicial z. Tu tarea es encontrar, por cada simulación, la fuerza del héroe cuando el juego termina.

Detalles de implementación

Debes implementar las siguientes funciones:

```
void init(int n, int[] s, int[] p, int[] w, int[] l)
```

- n: número de oponentes
- s, p, w, l: arreglos de tamaño n. Para $0 \leq i \leq n-1$:
  - $s[i]$ es la fuerza del oponente i. Es también la fuerza ganada por el héroe después de haber ganado contra el oponente i.
  - $p[i]$ es la fuerza ganada por el héroe después de de haber perdido contra el oponente i.
  - $w[i]$ es el calabozo en el que entra el héroe después de haber ganado contra el oponente i.

    - $l[i]$ es el calabozo en el que entra el héroe después de haber perdido contra el oponente i.
- Esta función es llamada exactamente una vez, antes de cualquier llamada a `simulate` (ver abajo).

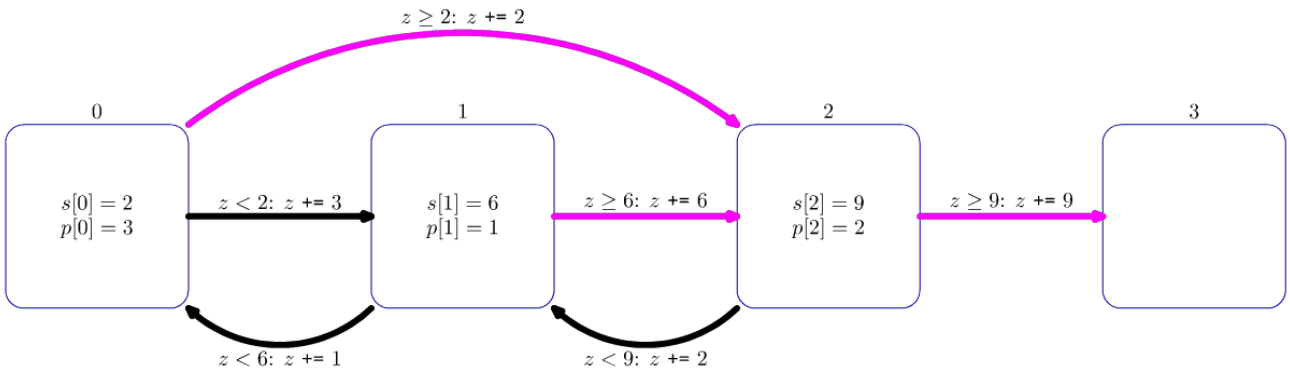
```
int64 simulate(int x, int z)
```

- x: el calabozo en el que el héroe entra primero.
- z: la fuerza inicial del héroe.
- Esta función deberá retornar la fuerza del héroe cuando el juego termine, asumiendo que el héroe inicia el juego entrando en el calabozo x con fuerza z.
- La función es llamada exactamente q veces.

Ejemplo

Considera la siguiente llamada:

```
init(3, [2, 6, 9], [3, 1, 2], [2, 2, 3], [1, 0, 1])
```



El diagrama anterior muestra la llamada. Cada cuadro muestra un calabozo. Para los calabozos 0, 1 y 2, los valores de $s[i]$ y $p[i]$ están indicados dentro de los cuadros. Las flechas magenta indican dónde se mueve el héroe después de haber ganado un enfrentamiento, mientras que las flechas negras indican dónde se mueve el héroe después de haber perdido.

Supongamos que el calificador llama `simulate(0, 1)`.

El juego procede de la siguiente manera:

Calabozo	Fuerza del héroe después del enfrentamiento	Resultado
0	1	Pierde
1	4	Pierde
0	5	Gana
2	7	Pierde
1	9	Gana
2	15	Gana
3	24	El juego termina

Entonces, la función deberá retornar 24.

Ahora supongamos que el calificador llama `simulate(2, 3)`.

El juego procede de la siguiente manera:

Calabozo	Fuerza del héroe después del enfrentamiento	Resultado
2	3	Pierde
1	5	Pierde
0	6	Gana
2	8	Pierde
1	10	Gana
2	16	Pierde
3	25	El juego termina

Entonces, la función deberá retornar 25.

Restricciones

- $1 \leq n \leq 400\,000$
- $1 \leq q \leq 50\,000$
- $1 \leq s[i], p[i] \leq 10^7$ (para todo $0 \leq i \leq n-1$)
- $0 \leq l[i], w[i] \leq n$ (para todo $0 \leq i \leq n-1$)
- $w[i] > i$ (para todo $0 \leq i \leq n-1$)
- $0 \leq x \leq n-1$
- $1 \leq z \leq 10^7$

Subtareas

1. (11 puntos) $n \leq 50\,000,\ q \leq 100,\ s[i], p[i] \leq 10\,000$ (para todo $0 \leq i \leq n-1$)

2. (26 puntos) $s[i] = p[i]$ (para todo $0 \leq i \leq n-1$)
3. (13 puntos) $n \leq 50\ 000$, todos los oponentes tienen la misma fuerza; es decir, $s[i] = s[j]$ para todo $0 \leq i, j \leq n-1$.
4. (12 puntos) $n \leq 50\ 000$, hay a lo sumo 5 valores distintos entre todos los valores de $s[i]$.
5. (27 puntos) $n \leq 50\ 000$
6. (11 puntos) Sin restricciones adicionales

Calificador de ejemplo

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

- línea 1: n q
- línea 2: $s[0]$ $s[1]$ $\ldots$ $s[n-1]$
- línea 3: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$
- línea 4: $w[0]$ $w[1]$ $\ldots$ $w[n-1]$
- línea 5: $l[0]$ $l[1]$ $\ldots$ $l[n-1]$
- línea $6+i$ ($0 \leq i \leq q-1$): x z para la i-ésima llamada a `simulate`.

El calificador de ejemplo imprime tus respuestas en el siguiente formato:

- línea $1+i$ ($0 \leq i \leq q-1$) : el valor de retorno de la i-ésima llamada a `simulate`.