Juego de Celdas

Robert está diseñando un nuevo juego de ordenador. El juego incluye un héroe, n oponentes y $n+1$ celdas. Los oponentes están numerados de 0 a $n-1$ y las celdas están numeradas de 0 a n. El oponente i ($0 \le i \le n-1$) se encuentra en la celda i y tiene una fuerza $s[i]$. No hay ningún oponente en la celda n.

El héroe comienza entrando en la celda x, con una fuerza z. Cada vez que el héroe entra en cualquier celda i ($0 \le i \le n-1$), se enfrenta al oponente i, y ocurre una de las siguientes cosas:

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

Observe que $p[i]$ puede ser menor, igual o mayor que $s[i]$. Además, $l[i]$ puede ser menor, igual o mayor que i. Independientemente del resultado del enfrentamiento, el oponente permanece en la celda i y mantiene la fuerza $s[i]$.

El juego termina cuando el héroe entra en la celda n. Se puede demostrar que el juego termina después de un número finito de enfrentamientos, independientemente de la celda inicial del héroe y de su fuerza.

Robert le ha pedido que pruebe su juego realizando q simulaciones. Para cada simulación, Robert define una celda inicial x y una fuerza inicial z. Tu tarea es averiguar, para cada simulación, la fuerza del héroe cuando el juego termina.

Detalles de la implementación

Usted deberá implementar los siguientes procedimientos:

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

- n: número de oponentes.
- s, p, w, l: matrices de longitud n. Para $0 \le i \le n-1$:
  - $s[i]$ es la fuerza del oponente i. También es la fuerza obtenida por el héroe después de ganar contra el oponente i.
  - $p[i]$ es la fuerza obtenida por el héroe después de perder contra el oponente i.
  - $w[i]$ es la celda en la que entra el héroe después de ganar contra el oponente i.
  - $l[i]$ es la celda en la que entra el héroe después de perder contra el oponente i.

- Este procedimiento se llama exactamente una vez, antes de cualquier llamada a `simulate` (ver más abajo).

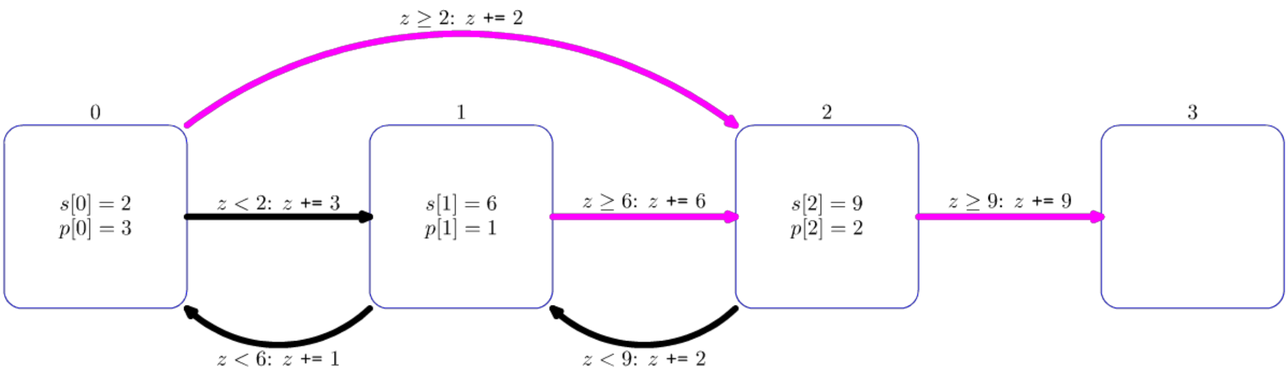
```
int64 simulate(int x, int z)
```

- x: la celda en la que el héroe entra primero.
- z: la fuerza inicial del héroe.
- Este procedimiento debería devolver la fuerza del héroe cuando el juego termina, asumiendo que el héroe comienza el juego entrando en la celda x, teniendo la fuerza z.
- El procedimiento es llamado exactamente q veces.

Ejemplo

Considere la siguiente llamada:

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



El diagrama anterior ilustra esta llamada. Cada cuadrado muestra una celda. Para las celdas 0, 1 y 2, los valores $s[i]$ y $p[i]$ se indican dentro de los cuadrados. Las flechas magenta indican dónde se mueve el héroe después de ganar un enfrentamiento, mientras que las flechas negras indican dónde se mueve el héroe después de perder.

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

El juego procede como sigue:

Celda	Fuerza del héroe antes de la confrontación	Resultado
0	1	Pierde
1	4	Pierde
0	5	Gana
2	7	Pierde
1	9	Gana
2	15	Gana
3	24	Fin del juego

Como tal, el procedimiento debería devolver 24.

Digamos que el calificador llama a `simulate(2, 3)`.

El juego procede como sigue:

Celda	Fuerza del héroe antes de la confrontación	Resultado
2	3	Pierde
1	5	Pierde
0	6	Gana
2	8	Pierde
1	10	Gana
2	16	Gana
3	25	Fin del juego

Como tal, el procedimiento debería devolver 25.

Restricciones

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

Subtareas

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

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, en otras palabras, $s[i] = s[j]$ 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) No hay restricciones adicionales.

Ejemplo del calificador

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 el i-th al llamar 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-th llamada a `simulate`.