Attraversa il dungeon!

Robert sta sviluppando un nuovo gioco in cui un eroe deve attraversare un dungeon di $n+1$ stanze (numerate da 0 a n). Ogni stanza i ($0 \leq i \leq n-1$) contiene un avversario di forza $s[i]$, mentre non ci sono avversari nella stanza n finale.

L'eroe inizia nella stanza x con forza z, e ogni volta che entra in una stanza i si scontra con l'avversario corrispondente, per cui:

- Se la forza dell'eroe è **maggiore o uguale** di quella dell'avversario $s[i]$, l'eroe vince e **aumenta** la sua forza sommandoci quella dell'avversario $s[i]$ ($s[i] \geq 1$); poi entra nella stanza $w[i]$ (con $w[i] > i$).
- Se invece la sua forza è **minore**, l'eroe perde e **aumenta** la sua forza di una quantità data $p[i]$ ($p[i] \geq 1$); poi entra nella stanza $l[i]$. Non ci sono restrizioni sui valori di $p[i]$ e $l[i]$ rispetto a $s[i]$ ed i.

Indipendentemente dal risultato, l'avversario della stanza i rimane nella sua stanza con la stessa forza $s[i]$.

Il gioco finisce quando l'eroe entra nella stanza n, e si può dimostrare che finisce sempre dopo un numero finito di scontri indipendentemente dagli x e z iniziali.

Devi eseguire q simulazioni del gioco, ciascuna con un suo valore x e z di partenza, e riportare per ognuna il livello di forza raggiunto dall'eroe al termine del gioco.

Note di implementazione

Devi implementare le seguenti funzioni:

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

- n: numero di avversari.
- s, p, w, l: vettori di lunghezza n, dove per ogni $0 \leq i \leq n-1$:
  - $s[i]$ è la forza dell'avversario i (e anche la forza guadagnata battendolo);
  - $p[i]$ è la forza guadagnata dall'eroe se ne viene invece sconfitto;
  - $w[i]$ è la stanza in cui si sposta l'eroe se vince;
  - $l[i]$ è la stanza in cui si sposta l'eroe se perde.
- Questa funzione è chiamata esattamente una volta, prima di tutte le chiamate a `simulate`.

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

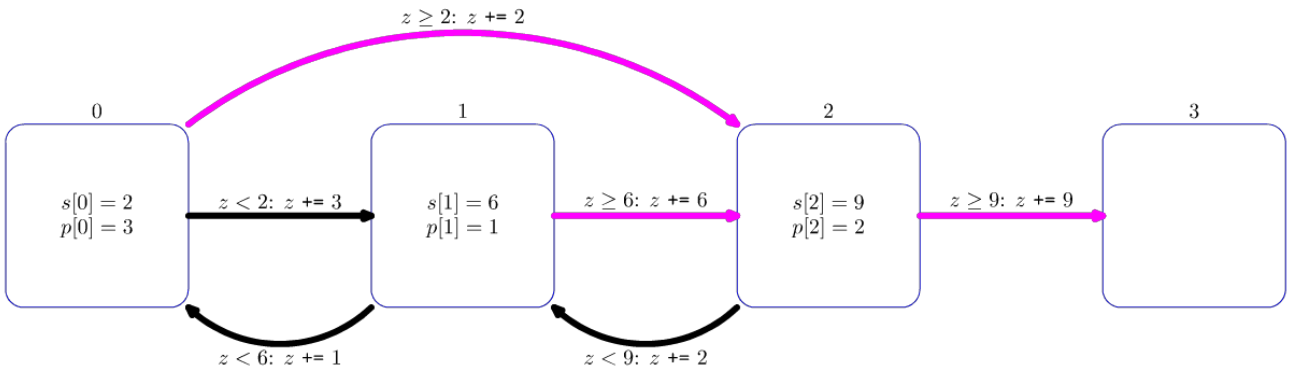
- x: la stanza in cui inizia l'eroe.
- z: la forza di partenza dell'eroe.
- Questa funzione deve restituire la forza finale dell'eroe al termine del gioco specificato.
- Questa funzione viene chiamata esattamente q volte.

Esempio

Considera la seguente chiamata:

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

Il diagramma seguente illustra questa chiamata:



Ogni quadrato descrive una stanza con i valori $s[i]$ e $p[i]$ (eccetto per la stanza finale). Le frecce magenta indicano dove si muove l'eroe se vince, quelle nere indicano dove si muove se perde.

Supponiamo che il grader chiami `simulate(0, 1)`. Il gioco procede come segue:

Stanza	Forza dell'eroe prima dello scontro	Risultato
0	1	sconfitta
1	4	sconfitta
0	5	vittoria
2	7	sconfitta
1	9	vittoria
2	15	vittoria
3	24	fine del gioco

Quindi la funzione deve restituire 24.

Supponiamo ora che il grader chiami `simulate(2, 3)`. Il gioco procede come segue:

Stanza	Forza dell'eroe prima dello scontro	Risultato
2	3	sconfitta
1	5	sconfitta
0	6	vittoria
2	8	sconfitta
1	10	vittoria
2	16	vittoria
3	25	fine del gioco

Quindi la funzione deve restituire 25.

Assunzioni

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

Subtask

1. (11 punti) $n \leq 50\,000$, $q \leq 100$, $s[i], p[i] \leq 10\,000$ (per ogni $0 \leq i \leq n-1$)
2. (26 punti) $s[i] = p[i]$ (per ogni $0 \leq i \leq n-1$)
3. (13 punti) $n \leq 50\,000$ e tutti gli avversari hanno la stessa forza: $s[i] = s[j]$ per ogni $0 \leq i, j \leq n-1$.
4. (12 punti) $n \leq 50\,000$ e ci sono al massimo 5 diversi valori per gli $s[i]$.
5. (27 punti) $n \leq 50\,000$.
6. (11 punti) Nessuna limitazione aggiuntiva.

Grader di esempio

Il grader di esempio legge l'input nel seguente formato:

- riga 1: n q
- riga 2: $s[0]$ $s[1]$ $\ldots$ $s[n-1]$
- riga 3: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$
- riga 4: $w[0]$ $w[1]$ $\ldots$ $w[n-1]$
- riga 5: $l[0]$ $l[1]$ $\ldots$ $l[n-1]$
- righe $6+i$ ($0 \le i \le q-1$): x z per la i-esima chiamata a `simulate`.

Il grader di esempio stampa l'output nel seguente formato:

- righe $1+i$ ($0 \le i \le q-1$) : il valore restituito dalla i-esima chiamata a `simulate`.