Dungeons Game

Robert este dezvoltatorul unui nou joc de calculator. Jocul presupune existența unui erou, a n adversari și $n+1$ temnițe. Adversarii sunt numerotați de la 0 la $n-1$ iar temnițele de la 0 la n. Adversarul i ($0 \leq i \leq n-1$) se află în temnița i și are puterea $s[i]$. Nu este nici un adversar în temnița n.

Eroul începe prin a intra în temnița x, cu puterea z. De fiecare dată când eroul intră într-o oarecare temniță i ($0 \leq i \leq n-1$), el se luptă cu adversarul i, și apare una din următoarele situații:

- Dacă puterea eroului este mai mare sau egală cu puterea adversarului $s[i]$, eroul învinge. Aceasta produce o **creștere** a puterii eroului cu $s[i]$ ($s[i] \geq 1$). În acest caz următoarea temniță în care intră eroul este $w[i]$ ($w[i] > i$).
- În caz contrar eroul pierde. Aceasta produce o **creștere** a puterii eroului cu $p[i]$ ($p[i] \geq 1$). În acest caz următoarea temniță în care intră eroul este $l[i]$.

De remarcat că $p[i]$ poate fi mai mic decât, egal cu, sau mai mare decât $s[i]$. De asemenea, $l[i]$ poate fi mai mic decât, egal cu, sau mai mare decât i. Indiferent de rezultatul confruntării, adversarul rămâne în temnița i și își menține puterea $s[i]$.

Jocul ia sfârșit când eroul intră în temnița n. Se poate demonstra că jocul se termină după un număr finit de confruntări, indiferent de la care temniță eroul începe jocul și de puterea inițială a eroului.

Robert te roagă să testezi jocul lui rulând q simulări. Pentru fiecare simulare, Robert definește temnița de start x și puterea inițială z. Sarcina ta este să determini, pentru fiecare simulare, puterea eroului la finalul jocului.

Detalii de Implementare

Trebuie să implementezi următoarele proceduri:

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

- n: numărul de adversari.
- s, p, w, l: tablouri unidimensionale de lungimea n. Pentru $0 \leq i \leq n-1$:
  - $s[i]$ este puterea adversarului i. Este de asemenea puterea obținută de erou după victoria asupra adversarului i.
  - $p[i]$ puterea obținută de erou după pierderea în fața adversarului i.
  - $w[i]$ este temnița în care intră eroul după victoria asupra adversarului i.
  - $l[i]$ este temnița în care intră eroul după pierderea în fața adversarului i.

- Această procedură este apelată exact odată, înainte de oricare din apelurile `simulate` (vezi mai jos).

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

- x: temnița de la care eroul începe jocul.
- z: puterea inițială a eroului.
- Procedura va returna puterea eroului la sfârșitul jocului, în presupunerea că jocul începe prin intrarea eroului în temnița x, având puterea z.
- Procedura este apelată exact q ori.

Exemplu

Considerăm următorul apel:

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

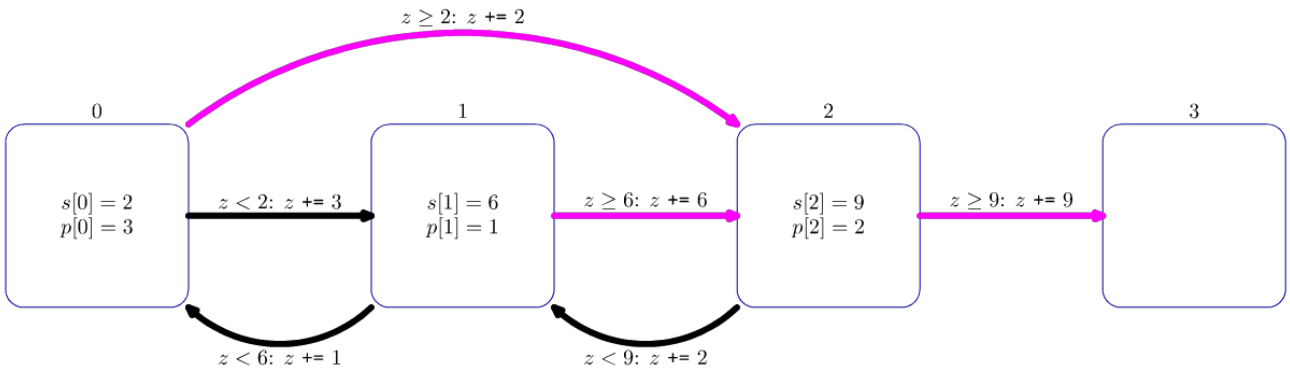

Diagrama de mai sus ilustrează apelul. Fiecare pătrat indică o temniță. Pentru temnițele 0, 1 și 2, valorile $s[i]$ și $p[i]$ sunt indicate în interiorul pătratelor. Săgețile de culoare magenta indică încotro se mută eroul după ce învinge într-o luptă, în timp ce săgețile negre indică unde merge eroul după înfrângeri.

Să zicem că grader-ul apelează `simulate(0, 1)`.

Jocul derulează după cum urmează:

Temnița	Puterea eroului până la luptă	Rezultat
0	1	Pierdere
1	4	Pierdere
0	5	Câștig
2	7	Pierdere
1	9	Câștig
2	15	Câștig
3	24	Sfârșit joc

Astfel, procedura trebuie să returneze 24.

Să zicem că grader-ul apelează `simulate(2, 3)`.

Jocul derulează după cum urmează:

Temnița	Puterea eroului până la luptă	Rezultat
2	3	Pierdere
1	5	Pierdere
0	6	Câștig
2	8	Pierdere
1	10	Câștig
2	16	Câștig
3	25	Sfârșit joc

Astfel, procedura trebuie să returneze 25.

Restricții

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

Subtaskuri

1. (11 puncte) $n \leq 50\ 000,\ q \leq 100,\ s[i], p[i] \leq 10\ 000$ (pentru oricare $0 \leq i \leq n-1$)

2. (26 puncte) $s[i] = p[i]$ (pentru oricare $0 \leq i \leq n-1$)
3. (13 puncte) $n \leq 50\ 000$, toți adversarii au aceeași putere, cu alte cuvinte, $s[i] = s[j]$ pentru oricare $0 \leq i, j \leq n-1$.
4. (12 puncte) $n \leq 50\ 000$, există cel mult 5 valori distincte printre valorile $s[i]$.
5. (27 puncte) $n \leq 50\ 000$
6. (11 puncte) Fără restricții suplimentare.

Exemplul de grader

Grader-ul citește datele de intrare în următorul format:

- linia 1: n q
- linia 2: $s[0]$ $s[1]$ $\ldots$ $s[n-1]$
- linia 3: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$
- linia 4: $w[0]$ $w[1]$ $\ldots$ $w[n-1]$
- linia 5: $l[0]$ $l[1]$ $\ldots$ $l[n-1]$
- linia $6+i$ ($0 \leq i \leq q-1$): x z pentru a i-ulea apel al `simulate`.

Grader-ul afișează răspunsul în următorul format:

- linia $1+i$ ($0 \leq i \leq q-1$) : valoarea returnată de al i-ulea apel al `simulate`.