Gra w Lochy (Dungeons Game)

Robert tworzy nową grę komputerową. W grze jest jeden bohater, n przeciwników oraz $n+1$ lochów. Przeciwnicy są ponumerowani od 0 do $n-1$, natomiast lochy są ponumerowane od 0 do n. Przeciwnik i ($0 \leq i \leq n-1$) znajduje się w i-tym lochu oraz ma siłę $s[i]$. Nie ma żadnych przeciwników w n-tym lochu.

Bohater zaczyna od x-tego lochu posiadając siłę z. Za każdy razem, gdy bohater wchodzi do i-tego lochu ($0 \leq i \leq n-1$), konfrontuje się z i-tym przeciwnikiem i zachodzi jeden z poniższych przypadków:

- Jeżeli siła bohatera jest większa lub równa sile przeciwnika $s[i]$, to bohater wygrywa. Sprawia to, że siła bohatera **zwiększa** się o $s[i]$ ($s[i] \geq 1$). W tym przypadku bohater następnie wchodzi do $w[i]$-tego lochu ($w[i] > i$).
- W przeciwnym przypadku, bohater przegrywa. Sprawia to, że siła bohatera **zwiększa** się o $p[i]$ ($[i] \geq 1$). W tym przypadku bohater następne wchodzi do $l[i]$-tego lochu.

Zwróć uwagę na to, że $p[i]$ może być mniejsze, równe albo większe niż $s[i]$. Podobnie, $l[i]$ może być mniejsze, równe albo większe niż i. Niezależnie od wyniku konfrontacji, przeciwnik zostaje w i-tym lochu oraz zachowuje siłę $s[i]$.

Gra się skończy, gdy bohater dotrze do n-tego lochu. Można udowodnić, że zawsze gra się skończy po skończonej liczbie ruchów, nieważne gdzie bohater zaczyna lub jaka jest jego siła początkowa.

Robert poprosił Cię o sprawdzenie jego gry poprzez wykonanie q symulacji. Dla każdej symulacji, Robert definiuje początkowy loch x oraz początkową siłę z. Twoim zadaniem jest sprawdzenie, dla każdej symulacji, jaka jest siła bohatera po skończeniu gry.

Szczegóły implementacyjne

Powinieneś zaimplementować następujące procedury:

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

- n: liczba przeciwników
- s, p, w, l: tablice długości n. Dla $0 \leq i \leq n-1$:
  - $s[i]$ to siła i-tego przeciwnika. Jest to również siła zyskana przez bohatera po pokonaniu i-tego przeciwnika.
  - $p[i]$ to siła zyskana przez bohatera po przegraniu z i-tym przeciwnikiem.
  - $w[i]$ to loch, do którego wchodzi bohater po wygraniu z i-tym przeciwnikiem.
  - $l[i]$ to loch, do którego wchodzi bohater po przegraniu z i-tym przeciwnikiem.

- Ta procedura jest wywołana dokładnie raz, przed jakimkolwiek wywołaniem `simulate` (patrz niżej).

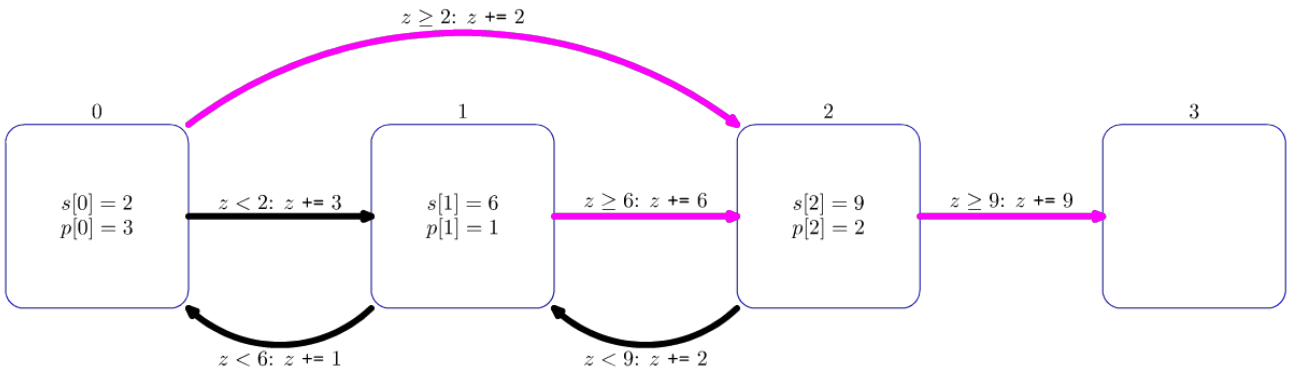
```
int64 simulate(int x, int z)
```

- x: loch, do którego bohater wchodzi na początku gry.
- z: siła początkowa bohatera.
- Ta funkcja powinna zwrócić siłę bohatera po zakończeniu gry, zakładając, że bohater zaczyna grę wchodząc do x-tego lochu z siłą z.
- Ta funkcja jest wywołana dokładnie q razy.

Przykład

Rozważ następujące wywołanie:

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



Powyższy diagram ilustruje to wywołanie. Każdy kwadrat przedstawia loch. Dla lochów 0, 1 oraz 2, wartości $s[i]$ oraz $p[i]$ są wskazane wewnątrz kwadratów. Różowe strzałki przedstawiają gdzie bohater się znajdzie po wygraniu konfrontacji, natomiast czarne strzałki przedstawiają gdzie bohater się znajdzie po przegraniu konfrontacji.

Powiedzmy, że sprawdzaczka wywoła `simulate(0, 1)`.

Gra przebiega w następujący sposób:

Loch	Siła bohatera przed konfrontacją	Wynik
0	1	Przegrana
1	4	Przegrana
0	5	Wygrana
2	7	Przegrana
1	9	Wygrana
2	15	Wygrana
3	24	Koniec gry

Wynikiem wywołania powinno więc być 24.

Powiedzmy, że sprawdzaczka wywoła `simulate(2, 3)`.

Gra przebiega w następujący sposób:

Loch	Siła bohatera przed konfrontacją	Wynik
2	3	Przegrana
1	5	Przegrana
0	6	Wygrana
2	8	Przegrana
1	10	Wygrana
2	16	Wygrana
3	25	Koniec gry

Wynikiem wywołania powinno więc być 25.

Ograniczenia

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

Podzadania

1. (11 punktów) $n \leq 50\,000,\ q \leq 100,\ s[i], p[i] \leq 10\,000$ (dla każdego $0 \leq i \leq n-1$)

2. (26 punktów) $s[i] = p[i]$ (dla każdego $0 \leq i \leq n-1$)
3. (13 punktów) $n \leq 50\ 000$, każdy przeciwnik ma taką samą siłę, czyli $s[i] = s[j]$ dla każdego $0 \leq i, j \leq n-1$.
4. (12 punktów) $n \leq 50\ 000$, jest co najwyżej 5 różnych wartości wśród wszystkich wartości $s[i]$.
5. (27 punktów) $n \leq 50\ 000$
6. (11 punktów) Brak dodatkowych ograniczeń.

Przykładowa sprawdzaczka

Przykładowa sprawdzaczka wczytuje wejście w następującym formacie:

- wiersz 1: $n\ \ q$
- wiersz 2: $s[0]\ \ s[1]\ \ \ldots\ \ s[n-1]$
- wiersz 3: $p[0]\ \ p[1]\ \ \ldots\ \ p[n-1]$
- wiersz 4: $w[0]\ \ w[1]\ \ \ldots\ \ w[n-1]$
- wiersz 5: $l[0]\ \ l[1]\ \ \ldots\ \ l[n-1]$
- wiersz $6+i$ ($0 \leq i \leq q-1$): $x\ \ z$ dla i-tego wywołania `simulate`.

Przykładowa sprawdzaczka wypisuje Twoje wyniki w następującym formacie:

- wiersz $1+i$ ($0 \leq i \leq q-1$) : wynik i-tego wywołania `simulate`.