Dungeons Game

O Robert está a desenhar um novo jogo de computador. O jogo envolve um herói, n adversários e $n+1$ masmorras. Os adversários são numerados de 0 a $n-1$ e as masmorras são numeradas de 0 a n. O adversário i ($0 \leq i \leq n-1$) está localizado na masmora i e tem força $s[i]$. Não existe nenhum adversário na masmorra n.

O herói começa com força z e entra na masmorra x. De cada vez que o herói entra numa qualquer masmorra i ($0 \leq i \leq n-1$), ele confronta o adversário i e acontece uma das duas coisas seguintes:

- Se a força do herói é maior ou igual à força do adversário $s[i]$, o herói ganha. Isto faz com que a força do herói **aumente** $s[i]$ ($s[i] \geq 1$) unidades. Neste caso o herói entra na masmorra $w[i]$ a seguir ($w[i] > i$).
- Caso contrário, o herói perde. Isto faz com que a força do herói **aumente** $p[i]$ ($p[i] \geq 1$) unidades. Neste caso o herói entra na masmorra $l[i]$ a seguir.

Nota que $p[i]$ pode ser menor, igual ou maior que $s[i]$. Nota também que $l[i]$ pode ser menor, igual ou maior que i. Seja qual for o desfecho do confronto, o adversário permanece na masmorra i e mantém uma força $s[i]$.

O jogo termina quando o herói entra na masmorra n. Pode ser mostrado que o jogo termina depois de um número finitio de confrontos, independentemente da masmorra inicial e força inicial do herói.

O Robert pediu-te para testar o jogo fazendo q simulações. Para cada simulação, ele define uma masmorra inicial x e uma força inicial z. A tua tarefa é descobrir, para cada simulação, qual a força do herói quando o jogo termina.

Detalhes de Implementação

Deves implementar as seguintes funções:

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

- n: número de adversários.
- s, p, w, l: arrays de tamanho n. Para cada $0 \leq i \leq n-1$:
  - $s[i]$ é a força do adversário i. É também a quantidade de força ganha pelo herói depois de vencer o adversário i.
  - $p[i]$ é a força ganha pelo herói depois de perder contra o adversário i.
  - $w[i]$ é a masmorra em que o herói entra depois de vencer o adversário i.
  - $l[i]$ é a masmorra em que o herói entra depois de perder contra o adversário i.

- Esta função é chamada exatamente uma vez, antes de quaisquer chamadas a `simulate` (ver em baixo).

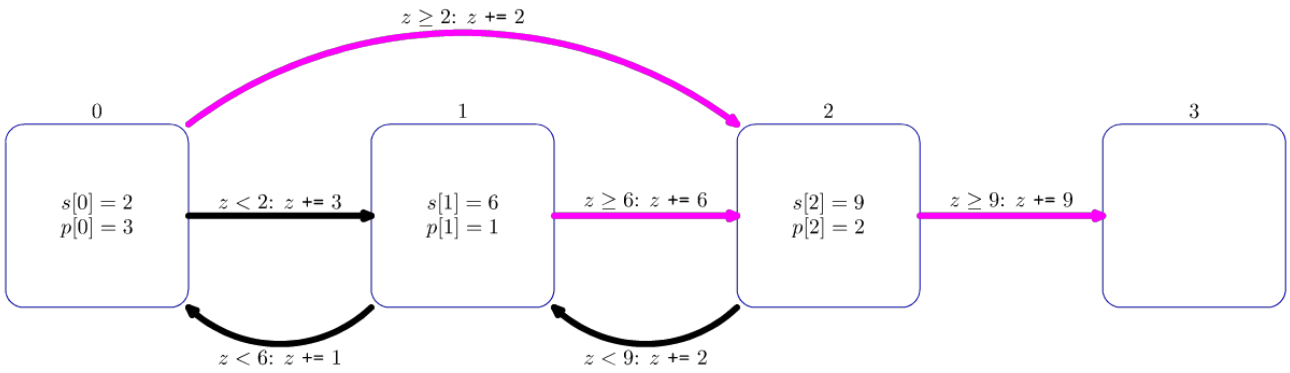
```
int64 simulate(int x, int z)
```

- x: a masmorra inicial do herói.
- z: a força inicial do herói.
- Esta função deve devolver a força do herói quando o jogo termina, assumindo que o herói começa o jogo com força z e na masmorra x.
- Esta função é chamada exatamente q vezes.

Exemplo

Considera a seguinte chamada:

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



O diagrama anterior ilustra esta chamada. Cada quadrado mostra uma masmora. Para as masmorras 0, 1 e 2, os valores $s[i]$ e $p[i]$ estão indicados dentro dos quadrados. As setas em magenta indicam para onde o herói se move depois de ganhar um confronto, ao passo que as setas a preto indicam para onde o herói se move depois de perder.

Imaginemos que o avaliador chama `simulate(0, 1)`.

O jogo desenrola-se da seguinte forma:

Masmorra	Força do herói antes do confronto	Resultado
0	1	Perde
1	4	Perde
0	5	Ganha
2	7	Perde
1	9	Ganha
2	15	Ganha
3	24	O jogo termina

Como tal, o procedimento deve devolver 24.

Imaginemos que o avaliador chama `simulate(2, 3)`.

O jogo desenrola-se da seguinte forma:

Masmorra	Força do herói antes do confronto	Resultado
2	3	Perde
1	5	Perde
0	6	Ganha
2	8	Perde
1	10	Ganha
2	16	Ganha
3	25	O jogo termina

Como tal, o procedimento deve devolver 25.

Restrições

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

Subtarefas

1. (11 pontos) $n \leq 50\ 000,\ q \leq 100,\ s[i], p[i] \leq 10\ 000$ (para $0 \leq i \leq n-1$)
2. (26 pontos) $s[i] = p[i]$ (para $0 \leq i \leq n-1$)
3. (13 pontos) $n \leq 50\ 000$, todos os adversários têm a mesma força, ou seja, $s[i] = s[j]$ para $0 \leq i, j \leq n-1$
4. (12 pontos) $n \leq 50\ 000$, existem no máximo 5 valores distintos entre todos os valores de $s[i]$
5. (27 pontos) $n \leq 50\ 000$
6. (11 pontos) Sem restrições adicionais.

Avaliador Exemplo

O avaliador exemplo lê o input no seguinte formato:

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

O avaliador exemplo escreve as tuas respostas no seguinte formato:

- linha $1+i$ ($0 \leq i \leq q-1$) : o valor de retorno da i-ésima chamada a `simulate`.