Jeu de Donjons

Robert est en train de créer un nouveau jeu vidéo. Le jeu met en scène un héro, n ennemis et $n+1$ donjons. Les ennemis sont numérotés de 0 à $n-1$ et les donjons sont numérotés de 0 à n. L'ennemi i ($0 \leq i \leq n-1$) se trouve dans le donjon i et a une force $s[i]$. Il n'y a pas d'ennemi dans le donjon n.

Le héro entre initialement dans le donjon x, avec une force de z. Chaque fois que le héro entre dans un donjon i ($0 \leq i \leq n-1$), il combat l'ennemi i, et l'une des choses suivantes se produit :

- Si la force du héro est supérieure ou égale à la force $s[i]$ de son ennemi, le héro l'emporte. La force du héro est alors **augmentée** de $s[i]$ ($s[i] \geq 1$). Dans ce cas, le prochain donjon dans lequel le héro entrera est $w[i]$ ($w[i] > i$).
- Sinon, le héro perd. La force du héro est alors **augmentée** de $p[i]$ ($p[i] \geq 1$). Dans ce cas, le prochain donjon dans lequel le héro entrera est $l[i]$.

Notez que $p[i]$ peut-être inférieur à, égal à, ou supérieur à $s[i]$. De même, $l[i]$ peut-être inférieur à, égal à, ou supérieur à i. Quel que soit le résultat de la confrontation, l'ennemi reste dans le donjon i et conserve une force de $s[i]$.

Le jeu se termine lorsque le héro entre dans le donjon n. Il est possible de démontrer que le jeu se termine après un nombre fini de confrontations, quelles que soient la position et la force initiales du héro.

Robert vous a demandé de tester son jeu en faisant q simulations. Pour chaque simulation, Robert définit un donjon de départ x et une force initiale z. Votre objectif est de déterminer, pour chaque simulation, la force du héro lorsque le jeu se termine.

Détails d'implémentation

Vous devez implémenter les fonctions suivantes :

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

- n : nombre d'ennemis.
- s, p, w, l : tableaux de taille n. Pour chaque $0 \leq i \leq n-1$:
  - $s[i]$ est la force de l'ennemi i. C'est aussi la force gagnée par le héro s'il l'emporte contre l'ennemi i.
  - $p[i]$ est la force gagnée par le héro s'il perd contre l'ennemi i.
  - $w[i]$ est le donjon dans lequel le héro entre après avoir gagné contre l'ennemi i.
  - $l[i]$ est le donjon dans lequel le héro entre après avoir perdu contre l'ennemi i.

- Cette fonction est appelée exactement une fois, avant tout appel à `simulate` (voir ci-dessous).

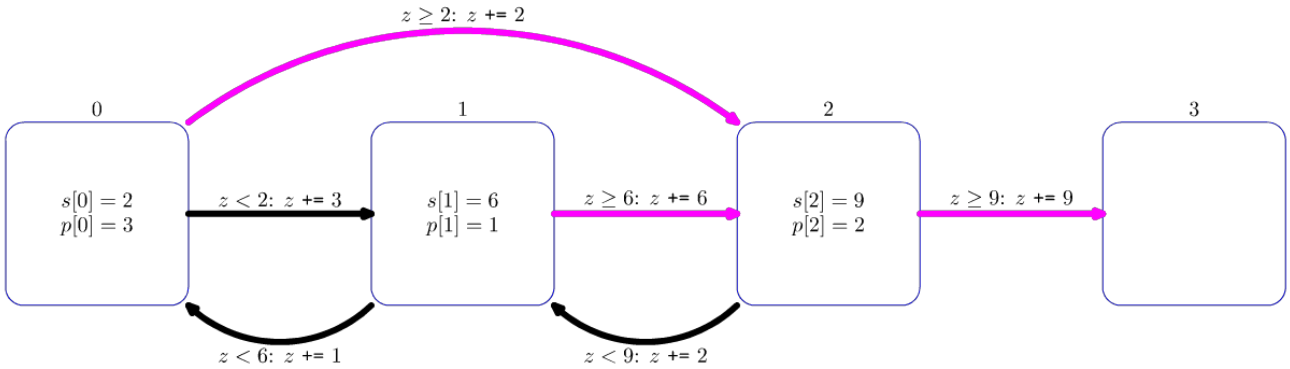
```
int64 simulate(int x, int z)
```

- x : le premier donjon dans lequel le héro entre.
- z : la force initiale du héro.
- Cette fonction doit renvoyer la force du héro lorsque le jeu se termine, en supposant que le héro commence au donjon x, avec une force de z.
- Cette fonction doit être appelée exactement q fois.

Exemple

Considérez l'appel suivant :

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



La figure ci-dessus illustre l'appel de fonction. Chaque carré représente un donjon. Dans les donjons 0, 1 et 2, les valeurs de $s[i]$ et $p[i]$ sont inscrites dans les carrés. Une flèche magenta indique où le héro se déplace après avoir gagné une confrontation, tandis qu'une flèche noire indique où le héro se déplace après une défaite.

Considérons que l'évaluateur fait l'appel `simulate(0, 1)`.

Le jeu se déroule comme suit :

Donjon	Force du héro avant confrontation	Résultat
0	1	Défaite
1	4	Défaite
0	5	Victoire
2	7	Défaite
1	9	Victoire
2	15	Victoire
3	24	Fin du jeu

Par conséquent, la fonction doit renvoyer 24.

Considérons que l'évaluateur fait l'appel `simulate(2, 3)`.

Le jeu se déroule comme suit :

Donjon	Force du héro avant confrontation	Résultat
2	3	Défaite
1	5	Défaite
0	6	Victoire
2	8	Défaite
1	10	Victoire
2	16	Victoire
3	25	Fin du jeu

Par conséquent, la fonction doit renvoyer 25.

Contraintes

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

Sous-tâches

1. (11 points) $n \leq 50\ 000,\ q \leq 100,\ s[i], p[i] \leq 10\ 000$ (pour tout $0 \leq i \leq n-1$)

2. (26 points) $s[i] = p[i]$ (pour tout $0 \leq i \leq n-1$)
3. (13 points) $n \leq 50\ 000$, tous les adversaires ont la même force, c'est-à-dire, $s[i] = s[j]$ pour tous $0 \leq i, j \leq n-1$.
4. (12 points) $n \leq 50\ 000$, il existe au plus 5 valeurs différentes parmi les valeurs des $s[i]$.
5. (27 points) $n \leq 50\ 000$
6. (11 points) Pas de contrainte supplémentaire.

Évaluateur d'exemple

L'évaluateur d'exemple lit l'entrée au format suivant :

- ligne 1 : n q
- ligne 2 : $s[0]$ $s[1]$ $\ldots$ $s[n-1]$
- ligne 3 : $p[0]$ $p[1]$ $\ldots$ $p[n-1]$
- ligne 4 : $w[0]$ $w[1]$ $\ldots$ $w[n-1]$
- ligne 5 : $l[0]$ $l[1]$ $\ldots$ $l[n-1]$
- ligne $6+i$ ($0 \leq i \leq q-1$) : x z pour le i-ème appel à `simulate`.

L'évaluateur d'exemple affiche vos réponses au format suivant :

- ligne $1+i$ ($0 \leq i \leq q-1$) : la valeur renvoyée par le i-ème appel à `simulate`.