ダンジョンゲーム (Dungeons Game)

Robert は新しいコンピューターゲームをデザインしている. このゲームには 1 人のヒーローと n 体の敵がいて, $n+1$ 個の地下牢がある. 敵には 0 から $n-1$ までの番号が付けられている. また, 地下牢には 0 から n までの番号が付けられている. 敵 i ($0 \leqq i \leqq n-1$) は地下牢 i におり, その強さは $s[i]$ である. 地下牢 n には敵はいない.

ヒーローははじめに, ある地下牢 x に強さ z の状態で入る. ヒーローが地下牢 i ($0 \leqq i \leqq n-1$) に入ると, 必ず敵 i と戦闘することになる. そして, 以下のいずれかのことが起こる.

- もし, ヒーローの強さが敵の強さ $s[i]$ 以上であれば, ヒーローが勝利する. その結果, ヒーローの強さが $s[i]$ ($s[i] \geqq 1$) だけ **増加する** . そして, ヒーローは次に地下牢 $w[i]$ ($w[i] > i$) に入ることになる.
- そうでなければ, ヒーローは敗北する. その結果, ヒーローの強さが $p[i]$ ($p[i] \geqq 1$) だけ **増加する** . そして, ヒーローは次に地下牢 $l[i]$ に入ることになる.

注意として, $p[i]$ は $s[i]$ より大きいかもしれないし, 小さいかもしれないし, 同じかもしれない. 同様に, $l[i]$ は i より大きいかもしれないし, 小さいかもしれないし, 同じかもしれない. 戦闘の勝敗にかかわらず, 敵は地下牢 i にいるままであり, 強さも $s[i]$ のままである.

このゲームはヒーローが地下牢 n に入った時点で終了する. ヒーローがはじめに, どの地下牢にどれだけの強さで入ったかにかかわらず, 有限回の戦闘でゲームが終了することを証明することができる.

Robert はゲームをテストするために, q 回のシミュレーションを実行することをあなたに依頼した. それぞれのシミュレーションにおいて Robert は, はじめにヒーローが入る地下牢 x とそのときのヒーローの強さ z を指定する. あなたはそれぞれのシミュレーションにおいて, ゲームが終了したときのヒーローの強さを求めなければならない.

実装の詳細

あなたは以下の関数を実装しなさい.

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

- n: 敵の数を表す.
- s, p, w, l: それぞれ長さ n の配列である. それぞれの i ($0 \leqq i \leqq n-1$) について:
  - $s[i]$ は敵 i の強さを表す. これはまた, ヒーローが敵 i に勝利したときに, ヒーローの強さがどれだけ増加するかも表す.
  - $p[i]$ はヒーローが敵 i に敗北したときに, ヒーローの強さがどれだけ増加するかを表す.
  - $w[i]$ はヒーローが敵 i に勝利したときに, ヒーローが次にどの地下牢に入るかを表す.
  - $l[i]$ はヒーローが敵 i に敗北したときに, ヒーローが次にどの地下牢に入るかを表す.

- この関数はちょうど 1 回呼び出される. この関数が呼び出される前に `simulate` 関数が呼び出されることはない.

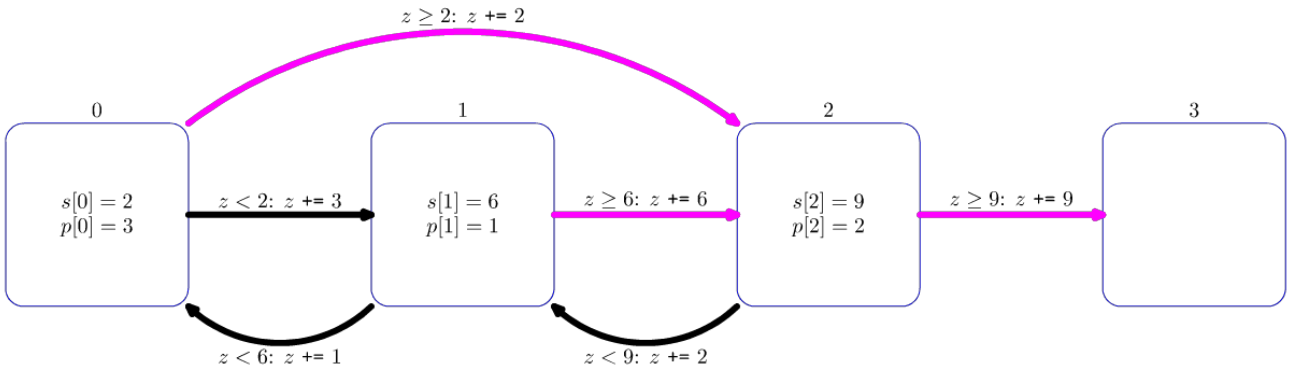
```
int64 simulate(int x, int z)
```

- x: ヒーローが初めに入る地下牢.
- z: ヒーローのはじめの強さ.
- この関数は, ヒーローがはじめに地下牢 x に強さ z の状態で入った場合の, ゲームが終了したときのヒーローの強さを返さなければならない.
- この関数はちょうど q 回呼び出される.

入出力例

以下の関数呼び出しを考える.

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



上の図は, この呼び出しを表す. それぞれの四角形は地下牢である. 地下牢 0, 1, 2 について, $s[i]$, $p[i]$ の値は四角形の中に示されている. マゼンタの矢印はヒーローが戦闘に勝利したときに次に入る地下牢を表し, 黒い矢印はヒーローが戦闘に敗北したときに次に入る地下牢を表す.

このとき, `simulate(0, 1)` が呼び出された場合を考える.

ゲームは以下のように進んでいく.

地下牢	戦闘前のヒーローの強さ	結果
0	1	敗北
1	4	敗北
0	5	勝利
2	7	敗北
1	9	勝利
2	15	勝利
3	24	ゲーム終了

以上より，この関数は 24 を返さなくてはならない．

次に，`simulate(2, 3)` が呼び出された場合を考える．

ゲームは以下のように進んでいく．

地下牢	戦闘前のヒーローの強さ	結果
2	3	敗北
1	5	敗北
0	6	勝利
2	8	敗北
1	10	勝利
2	16	勝利
3	25	ゲーム終了

以上より，この関数は 25 を返さなくてはならない．

制約

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

小課題

1. (11 点) $n \leqq 50\ 000$, $q \leqq 100$, $s[i], p[i] \leqq 10\ 000$ ($0 \leqq i \leqq n-1$)
2. (26 点) $s[i] = p[i]$ ($0 \leqq i \leqq n-1$)
3. (13 点) $n \leqq 50\ 000$, すべての敵は同じ強さである, すなわち $s[i] = s[j]$ ($0 \leqq i, j \leqq n-1$).
4. (12 点) $n \leqq 50\ 000$, $0 \leqq i \leqq n-1$ について $s[i]$ は高々 5 種類の値しか含まない.
5. (27 点) $n \leqq 50\ 000$
6. (11 点) 追加の制約はない.

採点プログラムのサンプル

採点プログラムのサンプルは以下の形式で入力を読み込む.

- 1 行目: n q
- 2 行目: $s[0]$ $s[1]$ $\ldots$ $s[n-1]$
- 3 行目: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$
- 4 行目: $w[0]$ $w[1]$ $\ldots$ $w[n-1]$
- 5 行目: $l[0]$ $l[1]$ $\ldots$ $l[n-1]$
- $6+i$ 行目 ($0 \leqq i \leqq q-1$): x z
  - i 番目の `simulate` 関数の呼び出しの引数

採点プログラムのサンプルは以下の形式であなたの答えを出力する.

- $1+i$ 行目 ($0 \leqq i \leqq q-1$): i 番目の `simulate` 関数の呼び出しの戻り値.