Chevaux

Comme ses ancêtres avant lui, Mansur adore élever des chevaux. Il a maintenant le plus grand troupeau du Kazakhstan, mais cela n'a pas toujours été le cas. Il y a N ans, Mansur n'était qu'un dzhigit (*jeune homme* en kazakh) et il n'avait qu'un seul cheval. Il rêvait de se faire beaucoup d'argent et de devenir un bai (*personne très riche* en kazakh).

Numérotons les années de 0 à $N-1$ dans l'ordre chronologique (c-à-d., l'année $N-1$ est la plus récente). La météo de chaque année a influencé la croissance du troupeau. Pour chaque année i, Mansur se souvient d'un coefficient de croissance (entier positif) $X[i]$. Si l'on commence l'année i avec h chevaux, on finit l'année avec un troupeau de $h \cdot X[i]$ chevaux.

Les chevaux ne peuvent être vendus qu'à la fin d'une année. Pour chaque année i, Mansur se souvient d'un nombre entier positif $Y[i]$: le prix de vente unitaire d'un cheval à la fin de l'année i. À la fin de chaque année, il est possible de vendre un nombre arbitraire de chevaux, chacun au même prix $Y[i]$.

Mansur se demande quel est le revenu le plus élevé qu'il aurait pu accumuler s'il avait choisi les meilleurs moments pour vendre ses chevaux pendant ces N années. Vous avez l'honneur d'être invité à la toi (*maison de vacances* en kazakh) de Mansur et il vous demande de répondre à cette question.

La mémoire de Mansur s'améliore au fur et à mesure de la soirée et il effectue M changements. Chaque changement modifiera soit une des valeurs $X[i]$, soit une des valeurs $Y[i]$. Après chaque changement, il vous redemande le revenu maximum qu'il aurait pu réaliser en vendant ses chevaux. Les changements de Mansur sont cumulatifs : chacune de vos réponses doit prendre en compte les changements précédents. Notez qu'un même $X[i]$ ou $Y[i]$ peut être changé plusieurs fois.

Les valeurs des réponses aux questions de Mansur peuvent être énormes. Afin d'éviter de travailler sur des grands nombres, vous devez fournir des réponses modulo $10^9 + 7$.

Exemple

Considérez qu'il y a $N = 3$ années, avec l'information suivante :

	0	**1**	**2**
X	2	1	3
Y	3	4	1

Pour ces valeurs initiales, Mansur peut maximiser son profit si il vend ses deux chevaux à la fin de l'année 1. La séquence complète est la suivante :

- Initialement, Mansur a 1 cheval.

- Après l'année 0, il aura $1 \cdot X[0] = 2$ chevaux.
- Après l'année 1, il aura $2 \cdot X[1] = 2$ chevaux.
- Il peut alors vendre ces deux chevaux. Le revenu total sera de $2 \cdot Y[1] = 8$.

Ensuite, supposons qu'il y a $M = 1$ changement : mettre $Y[1]$ à 2.

Après le changement, nous aurons :

	0	**1**	**2**
X	2	1	3
Y	3	2	1

Dans ce cas, une des solutions optimales est de vendre un cheval après l'année 0 et trois chevaux après l'année 2.

La séquence complète est la suivante :

- Initialement, Mansur a un cheval.
- Après l'année 0, il aura $1 \cdot X[0] = 2$ chevaux.
- Il peut vendre un de ces chevaux pour $Y[0] = 3$, il lui reste un cheval.
- Après l'année 1, il aura $1 \cdot X[1] = 1$ cheval.
- Après l'année 2, il aura $1 \cdot X[2] = 3$ chevaux.
- Il peut maintenant vendre ces trois chevaux pour $3 \cdot Y[2] = 3$. Le somme totale d'argent accumulée est $3 + 3 = 6$.

Tâche

On vous donne N, X, Y et la liste des changements. Avant le premier changement et après chaque changement, calculez le revenu maximal que Mansur pourrait tirer de ses chevaux, modulo $10^9 + 7$. Vous devez implémenter les fonctions `init`, `updateX` et `updateY`.

- `init(N, X, Y)` — sera appelé par l'évaluateur en premier et exactement une fois.
  - `N` : le nombre d'années.
  - `X` : un tableau de longueur N. Pour $0 \leq i \leq N - 1$, $X[i]$ donne le coefficient de croissance de l'année i.
  - `Y` : un tableau de longueur N. Pour $0 \leq i \leq N - 1$, $Y[i]$ donne le prix d'un cheval après l'année i.
  - Notez qu'à la fois `X` et `Y` spécifient les valeurs initiales données par Mansur (avant tout changement).
  - Après l'exécution de `init`, les tableaux `X` et `Y` restent valides et vous pouvez les modifier si vous le souhaitez.

    - La fonction doit retourner le revenu maximal que Mansur peut réaliser pour ces valeurs initiales, modulo $10^9 + 7$.

- `updateX(pos, val)`
    - `pos` : un entier dans l'intervalle $0, \ldots, N-1$.
    - `val` : la nouvelle valeur pour $X[\texttt{pos}]$.
    - La fonction doit retourner le revenu maximal que Mansur peut réaliser après ce changement, modulo $10^9 + 7$.
- `updateY(pos, val)`
    - `pos` : un entier dans l'intervalle $0, \ldots, N-1$.
    - `val` : la nouvelle valeur pour $Y[\texttt{pos}]$.
    - La fonction doit retourner le revenu maximal que Mansur peut réaliser après ce changement, modulo $10^9 + 7$.

Vous pouvez supposer que toutes les valeurs initiales ainsi que les valeurs après changement de $X[i]$ et $Y[i]$ sont comprises entre 1 et 10^9 inclus.
Après avoir appelé `init`, l'évaluateur appellera `updateX` et `updateY` plusieurs fois. Le nombre total d'appels à `updateX` et `updateY` sera M.

Sous-tâches

sous-tâche	points	N	M	contraintes supplémentaires
1	17	$1 \le N \le 10$	$M = 0$	$X[i], Y[i] \le 10$, $X[0] \cdot X[1] \cdot \ldots \cdot X[N-1] \le 1000$
2	17	$1 \le N \le 1000$	$0 \le M \le 1000$	aucune
3	20	$1 \le N \le 500000$	$0 \le M \le 100000$	$X[i] \ge 2$ et $val \ge 2$ pour respectivement `init` et `updateX`
4	23	$1 \le N \le 500000$	$0 \le M \le 10000$	aucune
5	23	$1 \le N \le 500000$	$0 \le M \le 100000$	aucune

Évaluateur fourni (grader)

L'évaluateur fourni lit son entrée à partir du fichier `horses.in` dans le format suivant :

- ligne 1 : `N`
- ligne 2 : `X[0] ... X[N - 1]`
- ligne 3 : `Y[0] ... Y[N - 1]`
- ligne 4 : `M`
- lignes 5, …, `M` + 4 : trois nombres `type pos val` (`type=1` pour `updateX` et `type=2` pour

`updateY`).

L'évaluateur fourni écrit la valeur de retour de `init` suivi des valeurs de retour de tous les appels à `updateX` et `updateY`.