Paarden

Mansur houdt van paardenfokken, net zoals zijn voorouders deden. Hij heeft nu de grootste kudde van Kazachstan. Maar dat was niet altijd zo. N jaar geleden was Mansur slechts een dzhigit (Kazachs voor *jongeman*) en had hij maar één paard. Hij droomde ervan veel geld te verdienen en het uiteindelijk tot bai te schoppen (Kazachs voor *een heel rijk persoon*)

Laat ons de jaren nummeren van 0 tot $N-1$ in chronologische volgorde (dus, jaar $N-1$ is het recentste). Het weer heeft elk jaar de groei van de kudde beïnvloed. Voor elk jaar i herinnert Mansur zich een positieve gehele groeicoëfficiënt $X[i]$. Als je jaar i begon met h paarden, dan eindigde je dat jaar met $h \cdot X[i]$ paarden in je kudde.

Paarden konden enkel verkocht worden op het eind van het jaar. Voor elk jaar i, herinnert Mansur zich een positieve integer $Y[i]$: de prijs waarvoor hij een paard kon verkopen aan het eind van jaar i. Na elk jaar was het mogelijk om willekeurig veel paarden te verkopen, elk aan dezelfde prijs $Y[i]$.

Mansur vraagt zich af wat de grootste hoeveelheid geld is die hij nu zou hebben als hij de best mogelijke momenten had gekozen om zijn paarden te verkopen gedurende deze N jaren. Je bent de eregast op Mansur's toi (Kazachs voor *vakantieverblijf*), en krijgt deze vraag voorgeschoteld.

Mansurs geheugen verbetert als de avond vordert, en dus maakt hij een reeks van M updates. Elke update verandert ofwel één van de waarden $X[i]$ of één van de waarden $Y[i]$. Na elke update vraagt hij je opnieuw wat de grootste hoeveelheid geld is die hij verdiend zou kunnen hebben in de paardenverkoop. Mansur's updates zijn cumulatief: elk van jouw antwoorden moet ook met alle vorige updates rekening houden. Let op dat een enkele $X[i]$ of $Y[i]$ verschillende keren geüpdated kan worden.

De juiste antwoorden op Mansurs vragen kunnen enorm zijn. Zodat je niet met grote getallen moet werken, hoef je de antwoorden enkel modulo $10^9 + 7$ terug te geven.

Voorbeeld

Stel dat er $N = 3$ jaren zijn, waarover je de volgende info hebt:

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

Voor deze beginwaarden, kan Mansur het meest verdienen als hij zijn twee paarden verkoopt aan het eind van jaar 1. Het hele proces ziet eruit als volgt:

- Mansur begint met 1 paard.
- Na jaar 0 heeft hij $1 \cdot X[0] = 2$ paarden.
- Na jaar 1 heeft hij $2 \cdot X[1] = 2$ paarden.
- Hij kan die twee paarden nu verkopen. De totale winst is dan $2 \cdot Y[1] = 8$.

Stel dan dat er $M = 1$ update is: verandering van $Y[1]$ naar 2.

Na deze update hebben we dan:

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

In dit geval is één van de optimale oplossingen om één paard te verkopen na jaar 0 en dan drie paarden na jaar 2. Het hele proces ziet eruit als volgt:

- Mansur begint met 1 paard.
- Na jaar 0 heeft hij $1 \cdot X[0] = 2$ paarden.
- Hij kan nu één van deze paarden verkopen voor $Y[0] = 3$, en heeft nog één paard over.
- Na jaar 1 heeft hij $1 \cdot X[1] = 1$ paard.
- Na jaar 2 heeft hij $1 \cdot X[2] = 3$ paarden.
- Hij kan nu deze drie paarden verkopen voor $3 \cdot Y[2] = 3$. De totale hoeveelheid geld is $3 + 3 = 6$.

Taak

Je krijgt N, X, Y, en de lijst met updates. Voor de eerste update, en na elke update, moet je de maximale hoeveelheid geld berekenen die Mansur zou verdiend kunnen hebben met zijn paarden, modulo $10^9 + 7$. Je moet de functies `init`, `updateX` en `updateY` implementeren.

- `init(N, X, Y)` — De grader roept deze functie eerst en exact één keer aan.
  - `N`: het aantal jaren.
  - `X`: een array van lengte N. Voor $0 \leq i \leq N - 1$, geeft $X[i]$ de groeicoëfficiënt jaar i.
  - `Y`: een array van lengte N. Voor $0 \leq i \leq N - 1$, geeft $Y[i]$ de prijs van een paard na jaar i.
  - Let op dat zowel `X` als `Y` de initiële waarden zijn, gegeven door Mansur (nog voor enige update).
  - Nadat `init` eindigt, blijven de arrays `X` en `Y` geldig, en mag je hun inhoud wijzigen als je dat wilt.
  - De functie moet de maximale hoeveelheid geld teruggeven die Mansur zou kunnen krijgen voor deze initiële waarden van X en Y, modulo $10^9 + 7$.

- `updateX(pos, val)`
  - `pos`: een integer in het interval $0, \ldots, N-1$ (inclusief).
  - `val`: de nieuwe waarde voor $X[\texttt{pos}]$.
  - De functie moet de maximale hoeveelheid geld teruggeven die Mansur zou kunnen krijgen na deze update, modulo $10^9 + 7$.
- `updateY(pos, val)`
  - `pos`: een integer in het interval $0, \ldots, N-1$ (inclusief).
  - `val`: de nieuwe waarde voor $Y[\texttt{pos}]$.
  - De functie moet de maximale hoeveelheid geld teruggeven die Mansur zou kunnen krijgen na deze update, modulo $10^9 + 7$.

Je mag aannemen dat alle initiële waarden, evenals alle geüpdatete waarden van $X[i]$ en $Y[i]$ liggen tussen 1 en 10^9 (inclusief).
Na het aanroepen van `init`, roept de grader verschillende keren `updateX` en `updateY` aan. Het totaal aantal aanroepen van `updateX` of `updateY` zal M zijn.

Subtaken

subtaak	punten	N	M	bijkomende beperkingen
1	17	$1 \leq N \leq 10$	$M = 0$	$X[i], Y[i] \leq 10$, $X[0] \cdot X[1] \cdot \ldots \cdot X[N-1] \leq 1,000$
2	17	$1 \leq N \leq 1,000$	$0 \leq M \leq 1,000$	geen
3	20	$1 \leq N \leq 500,000$	$0 \leq M \leq 100,000$	$X[i] \geq 2$ en $val \geq 2$ voor `init` en `updateX` respectievelijk
4	23	$1 \leq N \leq 500,000$	$0 \leq M \leq 10,000$	geen
5	23	$1 \leq N \leq 500,000$	$0 \leq M \leq 100,000$	geen

Voorbeeldgrader

De voorbeeldgrader leest input van het bestand `horses.in` in het volgende formaat:

- lijn 1: `N`
- lijn 2: `X[0] ... X[N - 1]`
- lijn 3: `Y[0] ... Y[N - 1]`
- lijn 4: `M`
- lijnen 5, …, `M` + 4: drie getallen `type pos val` (`type=1` voor `updateX` en `type=2` voor `updateY`).

De voorbeeldgrader print de returnwaarde van `init` gevolgd door de returnwaarden van alle aanroepen van `updateX` en `updateY`.