Speelgoedtrein (Toy Train)

Arezou en haar broer Borzou zijn een tweeling. Ze hebben voor hun verjaardag een geweldige speelgoedtrein gekregen, en ze hebben er een spoornetwerk mee gebouwd met n stations en m *eenrichtings*sporen. De stations zijn genummerd van 0 tot en met $n-1$. Elk spoor vertrekt uit één station en komt in hetzelfde of een ander station aan. Er vertrekt tenminste één spoor uit elk station.

Sommige stations zijn *laadstations*. Telkens de trein aankomt in een laadstation wordt hij volledig opgeladen. Een volledig opgeladen trein heeft genoeg energie om over n opeenvolgende sporen te rijden. Dus, de trein valt, net wanneer hij het $n+1$-ste spoor na de laatste oplaadbeurt oprijdt, zonder energie.

In elk station is er een wissel die naar eender welk uitgaand spoor kan wijzen. De trein verlaat het station via het spoor waarnaar de wissel wijst.

De tweeling gaat een spel spelen met hun trein. Ze hebben al alle stations onder elkaar verdeeld: elk station is ofwel van Arezou ofwel van Borzou. Er is maar één trein. In het begin van het spel staat de trein in station s en is hij volledig opgeladen. Om het spel te starten, wijst de eigenaar van station s de wissel van station s naar een van de uitgaande sporen. Vervolgens zetten ze de trein op en begint die langs de sporen te rijden.

Telkens als de trein voor de eerste keer een station binnenrijdt wijst de eigenaar van dat station de wissel van dat station naar een uitgaand spoor. Eens een wissel ergens naar wijst, blijft hij in dezelfde positie voor de rest van het spel. Zodus, als een trein een station opnieuw binnenkomt, zal het daar langs het zelfde spoor als voorheen terug weggaan.

Gezien er een eindig aantal stations is, zal de trein uiteindelijk een *lus* beginnen. Een lus is een opeenvolging van *verschillende* stations $c[0], c[1], \cdots, c[k-1]$ zodat de trein uit station $c[i]$ (met $0 \leq i < k-1$) vertrekt langs het spoor naar station $c[i+1]$, en hij vanuit station $c[k-1]$ over het spoor naar station $c[0]$ terugrijdt. Een lus kan slechts uit één station bestaat ($k = 1$) als er een spoor vertrekt en aankomt in hetzelfde station.

Arezou wint het spel als de trein eindeloos blijft doorgaan, en Borzou wint als de trein zonder energie valt. Anders gezegd, als er minstens één laadstation is onder de $c[0], c[1], \cdots, c[k-1]$ kan de trein eindeloos opladen en rondrijden, en wint Arezou. Anders zal de energie opraken (mogelijk na verschillende keren de lus rond te rijden), en wint Borzou.

Je krijgt de beschrijving van het spoornetwerk. Arezou en Borzou gaan n spelletjes spelen. In het s-de spelletje, voor $0 \leq s \leq n-1$, bevindt de trein zich initieel in station s. Jouw taak is om, voor elk spelletje, te bepalen of er een strategie bestaat voor Arezou die garandeert dat ze wint, onafhankelijk van hoe Borzou speelt.

Implementatiedetails

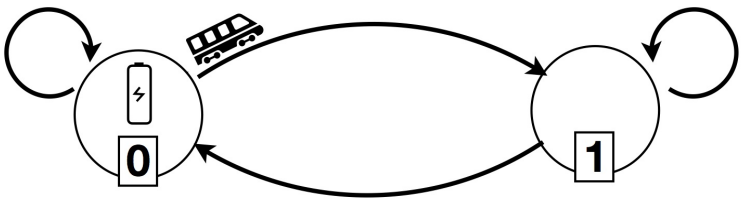
Je moet de volgende functie implementeren:

```
int[] who_wins(int[] a, int[] r, int[] u, int[]v)
```

- a: array van lengte n. Als station i van Arezou is geldt $a[i] = 1$. Anders is het station van Borzou en $a[i] = 0$.
- r: array van lengte n. Als station i een laadstation is geldt $r[i] = 1$, anders $r[i] = 0$.
- u en v: arrays van lengte m. Voor alle $0 \leq i \leq m - 1$ is er een eenrichtingsspoor vertrekkend vanuit station $u[i]$ en eindigend in station $v[i]$.
- Deze functie moet een array w van lengte n teruggeven. Voor alle $0 \leq i \leq n - 1$ moet de waarde van $w[i]$ 1 zijn als Arezou het spelletje dat begint in station i kan winnen, onafhankelijk van hoe Borzou speelt. Anders moet de waarde van $w[i]$ 0 zijn.

Voorbeeld

```
who_wins([0, 1], [1, 0], [0, 0, 1, 1], [0, 1, 0, 1])
```



- Er zijn 2 stations. Station 0 is van Borzou en is een laadstation. Station 1 is van Arezou en is geen laadstation.
- Er zijn 4 sporen $(0, 0), (0, 1), (1, 0)$ en $(1, 1)$, waarbij (i, j) aangeeft dat er een eenrichtingsspoor bestaat van station i naar station j.
- Beschouw het spel waarbij de trein in het begin in station 0 wordt geplaatst. Als Borzou de wissel van station 0 naar het spoor $(0, 0)$ wijst, rijdt de trein eindeloos in een lus langs dit spoor (merk op dat station 0 een laadstation is). In dit geval wint Arezou. Anders, als Borzou de wissel van station 0 naar spoor $(0, 1)$ wijst, kan Arezou de wissel van station 1 naar $(1, 0)$ wijzen. Als dit gebeurt zal de trein eindeloos door beide stations rijden. Arezou wint opnieuw, vermits station 0 een laadstation is en de trein nooit zal stoppen. Bijgevolg kan Arezou het spelletje winnen, onafhankelijk van wat Borzou doet.
- Met een analoge redenering kan Arezou ook winnen in het spelletje waarbij de trein in station 1 vertrekt, onafhankelijk hoe Borzou speelt. Zodus, de functie moet $[1, 1]$ teruggeven.

Beperkingen

- $1 \leq n \leq 5000$.
- $n \leq m \leq 20\,000$.

- Er is tenminste één laadstation.
- Er vertrekt tenminste één spoor vanuit elk station.
- Er kunnen sporen zijn die in hetzelfde station aankomen als waar ze vertrekken (dus, $u[i] = v[i]$).
- Elk spoor is uniek. Met andere woorden zijn er geen indices i en j ($0 \leq i < j \leq m - 1$) zodat $u[i] = u[j]$ en $v[i] = v[j]$.
- $0 \leq u[i], v[i] \leq n - 1$ (voor alle $0 \leq i \leq m - 1$).

Subtaken

1. (5 punten) Voor alle $0 \leq i \leq m - 1$, ofwel $v[i] = u[i]$ of $v[i] = u[i] + 1$.
2. (10 punten) $n \leq 15$.
3. (11 punten) Alle stations zijn van Arezou
4. (11 punten) Alle stations zijn van Borzou.
5. (12 punten) Er is exact één laadstation.
6. (51 punten) Geen bijkomende beperkingen.

Voorbeeldgrader

De voorbeeldgrader leest de input in het volgende formaat:

- Lijn 1: n m
- Lijn 2: $a[0]$ $a[1]$ $\ldots$ $a[n-1]$
- Lijn 3: $r[0]$ $r[1]$ $\ldots$ $r[n-1]$
- Lijn $4 + i$ (voor $0 \leq i \leq m - 1$): $u[i]$ $v[i]$

De voorbeeldgrader print de terugkeerwaarde van `who_wins` in het volgende formaat:

- Lijn 1: $w[0]$ $w[1]$ $\ldots$ $w[n-1]$