Train électrique (Toy Train)

Borzou et sa sœur jumelle Arezou ont reçu un magnifique kit de train électrique pour leur anniversaire. Ils ont construit un réseau ferré constitué de n gares et de m voies ferrées *à sens unique*. Les gares sont numérotées de 0 à $n-1$. Chaque voie part d'une station et arrive à une station, qui peut être la même que celle de départ. De chaque gare part au moins une voie.

Certaines gares sont des *gares de recharge*. Chaque fois qu'un train arrive à une gare de recharge, il reçoit une charge complète. Un train complètement chargé a assez d'énergie pour parcourir n voies consécutives, c'est-à-dire qu'il épuisera ses batteries et s'arrêtera exactement au moment d'emprunter la $n+1$-ième voie depuis sa dernière recharge.

À chaque gare se trouve un aiguillage qui peut être réglé sur n'importe laquelle des voies qui partent de cette gare. Le train emprunte la voie sur laquelle est réglé l'aiguillage.

Les jumeaux ont inventé un nouveau jeu avec leur train. Ils se sont répartis l'ensemble des gares : chaque gare est contrôlée soit par Arezou, soit par Borzou. Il y a un seul train. Au début d'une partie, le train se trouve dans la gare s et est complètement chargé. Lorsque le jeu commence, le joueur qui contrôle la gare s règle l'aiguillage de la gare s sur une des voies qui en partent. Ils démarrent ensuite le train, qui commence à rouler sur les voies.

Lorsque le train entre dans l'une des gares pour la première fois, le joueur qui contrôle cette gare règle l'aiguillage correspondant. Une fois l'aiguillage réglé, il reste dans la même position jusqu'à la fin de la partie. Ainsi, lorsqu'un train entre à nouveau dans une station déjà visitée, il la quitte par la même voie que lors de ses précédents passages.

Comme les gares sont en nombre fini, au bout d'un certain temps, le train va se retrouver dans un *cycle*. Un cycle est une suite de gares *distinctes* $c[0], c[1], \cdots, c[k-1]$ telle que l'aiguillage de la gare $c[i]$ $(0 \leq i < k)$ est dirigé vers une voie qui termine dans la gare $c[i+1]$, et l'aiguillage de la gare $c[k-1]$ est dirigé vers une voie qui termine dans la gare $c[0]$. Notez qu'un cycle peut être composé d'une unique gare (i.e. $k = 1$), si l'aiguillage de la gare $c[0]$ est dirigé vers la gare $c[0]$.

Arezou gagne la partie si le train continue à rouler indéfiniment, et Borzou gagne si le train se décharge complètement et s'arrête. Autrement dit, s'il y a au moins une gare de recharge parmi les gares $c[0], c[1], \cdots, c[k-1]$, le train peut se recharger et rouler indéfiniment, et donc Arezou gagne. Dans le cas contraire, le train va se décharger complètement (éventuellement après avoir parcouru le cycle plusieurs fois), et Borzou gagne.

On vous donne la description du réseau ferré. Arezou et Borzou vont jouer n parties. Pour la s-ième partie, $0 \leq s \leq n-1$, le train se trouve initialement dans la gare s. Vous devez, pour chaque partie, déterminer s'il existe une stratégie qui garantit la victoire à Arezou, quelle que soit la stratégie

de Borzou.

Détails d'implémentation

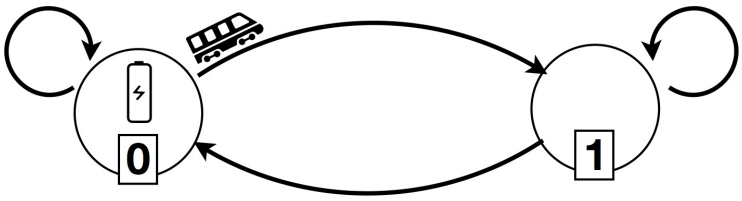
Vous devez implémenter la fonction suivante

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

- a : tableau de taille n. Si Arezou contrôle la gare i, $a[i] = 1$. Sinon, Borzou contrôle la gare i et $a[i] = 0$.
- r : tableau de taille n. Si la gare i est une gare de recharge, $r[i] = 1$. Sinon, $r[i] = 0$.
- u et v : tableaux de taille m. Pour tout $0 \leq i \leq m - 1$, il existe une voie à sens unique qui part de la gare $u[i]$ et arrive à la gare $v[i]$.
- Cette fonction doit renvoyer un tableau de taille w et de longueur n. Pour tout $0 \leq i \leq n - 1$, $w[i]$ doit être égal à 1 si Arezou peut gagner la partie qui commence à la gare i, quelle que soit la stratégie de Borzou. Dans le cas contraire, $w[i]$ doit être égal à 0.

Exemple

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



- Il y a 2 gares. Borzou contrôle la gare 0, qui est une gare de recharge. Arezou contrôle la gare 1, qui n'est pas une gare de recharge.
- Il y a 4 voies $(0, 0), (0, 1), (1, 0)$, et $(1, 1)$, où (i, j) représente une voie à sens unique de la gare i à la gare j.
- Considérons la partie où le train commence à la gare 0. Si Borzou règle l'aiguillage de la gare 0 sur la voie $(0, 0)$, le train va boucler indéfiniment sur cette voie (puisque la gare 0 est une gare de recharge). Dans ce cas, Arezou gagne la partie. Dans l'autre cas, où Borzou règle l'aiguillage de la gare 0 sur la voie $(0, 1)$, Arezou peut régler l'aiguillage de la gare 1 sur la voie $(1, 0)$. Cela fera boucler le train indéfiniment entre les deux gares, et Arezou gagne encore une fois. Ainsi, Arezou peut gagner, quelle que soit la stratégie de Borzou.
- Par un raisonnement similaire, pour la partie qui commence à la gare 1, Arezou peut également gagner quelle que soit la stratégie de Borzou. La fonction doit donc renvoyer $[1, 1]$.

Contraintes

- $1 \leq n \leq 5000$.

- $n \leq m \leq 20\,000$.
- Il y a au moins une gare de recharge.
- Il y a au moins une voie qui part de chaque gare.
- Il peut y avoir des voies qui ont la même gare de départ et d'arrivée (i.e. $u[i] = v[i]$).
- Toutes les voies sont distinctes. Autrement dit, il n'y aura jamais deux indices i et j $(0 \leq i < j \leq m - 1)$ tels que $u[i] = u[j]$ et $v[i] = v[j]$.
- $0 \leq u[i], v[i] \leq n - 1$ (pour tout $0 \leq i \leq m - 1$).

Sous-tâches

1. (5 points) Pour $0 \leq i \leq m - 1$, soit $v[i] = u[i]$, soit $v[i] = u[i] + 1$.
2. (10 points) $n \leq 15$.
3. (11 points) Arezou contrôle toutes les gares.
4. (11 points) Borzou contrôle toutes les gares.
5. (12 points) Il y a exactement une gare de chargement.
6. (51 points) Aucune contrainte additionnelle.

Évaluateur d'exemple

L'évaluateur d'exemple lit l'entrée dans le format suivant :

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

L'évaluateur d'exemple affiche la valeur de retour de `who_wins` dans le format suivant :

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