Llaves

Timothy el arquitecto ha diseñado un nuevo juego de escape. En este juego, hay n habitaciones numeradas de 0 a $n-1$. Inicialmente, cada habitación contiene exactamente una llave. Cada llave tiene un tipo, el cual es un entero entre 0 y $n-1$, inclusive. El tipo de llave en la habitación i ($0 \leq i \leq n-1$) es $r[i]$. Nota que multiples habitaciones pueden contener el mismo tipo de llave; es decir, los valores $r[i]$ no necesariamente son distintos.

Hay también m conectores **bidirecionales** en el juego, numerados de 0 a $m-1$. El conector j ($0 \leq j \leq m-1$) conecta una pareja de habitaciones diferentes $u[j]$ and $v[j]$. Una pareja de habitaciones puede ser conectada por multiples conectores.

En el juego hay un sólo jugador que recoge las llaves y se mueve entre las habitaciones atravesando los conectores. Se dice que el jugador **atraviesa** el conector j cuando usa el conector para moverse desde la habitación $u[j]$ hasta la habitación $v[j]$, o viceversa. El jugador sólo puede atravesar el conector j si ha recogido una llave de tipo $c[j]$ anteriormente.

En cualquier punto durante el juego, el jugador está en alguna habitación x y puede realizar dos tipos de acciones:

- recoger la llave en la habitación x, cuyo tipo es $r[x]$ (a no ser que ya la haya recogido),
- atravesar un conector j, donde ya sea $u[j] = x$ o $v[j] = x$, si el jugador ya ha recogido una llave de tipo $c[j]$ anteriormente. Nota que el jugador **nunca** deshecha una llave que haya recogido.

El jugador **inicia** el juego en alguna habitación s sin poseer llave alguna. Una habitación t es **accesible** desde una habitación s, si el jugador que inició el juego en la habitación s puede realizar alguna secuencia de acciones descritas anteriormente, y llegar a la habitación t.

Para cada habitación i ($0 \leq i \leq n-1$), sea $p[i]$ el número de habitaciones accesibles desde la habitación i. A Timothy le gustaría saber el conjunto de índices i que alcanzan el valor mínimo de $p[i]$ para $0 \leq i \leq n-1$.

Detalles de Implementación

Tienes que implementar la siguiente función:

```
int[] find_reachable(int[] r, int[] u, int[] v, int[] c)
```

- r: un arreglo de tamaño n. Para cada i ($0 \leq i \leq n-1$), la llave en la habitación i es de tipo $r[i]$.

- u, v: dos arreglos de tamaño m. Para cada j ($0 \leq j \leq m-1$), el conector j conecta las habitaciones $u[j]$ y $v[j]$.
- c: un arreglo de tamaño m. Para cada j ($0 \leq j \leq m-1$), el tipo de llave que se necesita para atravesar el conector j es $c[j]$.
- Esta función deberá retornar un arreglo a de tamaño n. Para cada $0 \leq i \leq n-1$, el valor de $a[i]$ deberá ser 1 si para cada j tal que $0 \leq j \leq n-1$, $p[i] \leq p[j]$. De lo contrario, el valor de $a[i]$ deberá ser 0.

Ejemplos

Ejemplo 1

Considere la siguiente llamada:

```
find_reachable([0, 1, 1, 2],
               [0, 0, 1, 1, 3], [1, 2, 2, 3, 1], [0, 0, 1, 0, 2])
```

Si el jugador inicia el juego en la habitación 0, puede realizar la siguiente secuencia de acciones:

Habitación actual	Acción
0	Recoger la llave de tipo 0
0	Atravesar el conector 0 hacia la habitación 1
1	Recoger la llave de tipo 1
1	Atravesar el conector 2 hacia la habitación 2
2	Atravesar el conector 2 hacia la habitación 1
1	Atravesar el conector 3 hacia la habitación 3

Por lo tanto la habitación 3 es accesible desde la habitación 0. Similarmente, se pueden construir secuencias monstrando que todas las habitaciones son accesibes desde la habitación 0, lo que implica $p[0] = 4$. La siguiente tabla muestra las habitaciones accesibles para todas las habitaciones iniciales:

Habitación inicial i	Habitaciones accesibles	$p[i]$
0	$[0, 1, 2, 3]$	4
1	$[1, 2]$	2
2	$[1, 2]$	2
3	$[1, 2, 3]$	3

El valor más pequeño de $p[i]$ para todas las habitaciones es 2, y esto sucede cuando $i = 1$ ó $i = 2$. Por lo tanto, la funcion deberá retornar $[0, 1, 1, 0]$.

Ejemplo 2

```
find_reachable([0, 1, 1, 2, 2, 1, 2],
               [0, 0, 1, 1, 2, 3, 3, 4, 4, 5],
               [1, 2, 2, 3, 3, 4, 5, 5, 6, 6],
               [0, 0, 1, 0, 0, 1, 2, 0, 2, 1])
```

La siguiente tabla muestra las habitaciones accesibles:

Habitación inicial i	Habitaciones accesibles	$p[i]$
0	$[0, 1, 2, 3, 4, 5, 6]$	7
1	$[1, 2]$	2
2	$[1, 2]$	2
3	$[3, 4, 5, 6]$	4
4	$[4, 6]$	2
5	$[3, 4, 5, 6]$	4
6	$[4, 6]$	2

El valor más pequeño de $p[i]$ para todas las habitaciones es 2, y sucede cuando $i \in \{1, 2, 4, 6\}$. Por lo tanto, la función deberá retornar $[0, 1, 1, 0, 1, 0, 1]$.

Ejemplo 3

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

La siguiente tabla muestra las habitaciones accesibles:

Habitación inicial i	Habitaciones accesibles	$p[i]$
0	$[0, 1]$	2
1	$[0, 1]$	2
2	$[2]$	1

El valor más pequeño de $p[i]$ para todas las habitaciones es 1, cuando $i = 2$. Por lo tanto, la función deberá retornar $[0, 0, 1]$.

Restricciones

- $2 \leq n \leq 300\,000$
- $1 \leq m \leq 300\,000$

- $0 \leq r[i] \leq n-1$ para todo $0 \leq i \leq n-1$
- $0 \leq u[j], v[j] \leq n-1$ y $u[j] \neq v[j]$ para todo $0 \leq j \leq m-1$
- $0 \leq c[j] \leq n-1$ para todo $0 \leq j \leq m-1$

Subtareas

1. (9 puntos) $c[j] = 0$ para todo $0 \leq j \leq m-1$ y $n, m \leq 200$
2. (11 puntos) $n, m \leq 200$
3. (17 puntos) $n, m \leq 2000$
4. (30 puntos) $c[j] \leq 29$ (para todo $0 \leq j \leq m-1$) y $r[i] \leq 29$ (para todo $0 \leq i \leq n-1$)
5. (33 puntos) Sin restricciones adicionales

Calificador Ejemplo

El calificador ejemplo lee la entrada en el siguiente formato:

- línea 1: n m
- línea 2: $r[0]$ $r[1]$ $\ldots$ $r[n-1]$
- línea $3+j$ ($0 \leq j \leq m-1$): $u[j]$ $v[j]$ $c[j]$

El calificador ejemplo imprime el valor de retorno de `find_reachable` en el siguiente formato:

- línea 1: $a[0]$ $a[1]$ $\ldots$ $a[n-1]$