Keys

Timothy el arquitecto ha diseñado un nuevo juego de escape. En este juego, existen n cuartos numerados de 0 hasta $n-1$. Inicialmente, cada cuarto 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 el cuarto i ($0 \leq i \leq n-1$) es $r[i]$. Note que múltiples cuartos pueden tener llaves del mismo tipo, es decir, los valores $r[i]$ no son necesariamente distintos.

Existen también m conectores **bidireccionales** en el juego, numerados de 0 a $m-1$. El conector j ($0 \leq j \leq m-1$) conecta un par de cuartos diferentes $u[j]$ y $v[j]$. Un par de cuartos puede estar conectado por múltiples conectores.

El juego es jugado por un solo jugador, el cual, recolecta las llaves y se mueve entre cuartos recorriendo los conectores. Decimos que el jugador **Recorre** el conector j cuando utiliza este conector para moverse del cuarto $u[j]$ al cuarto $v[j]$ o viceversa. El jugador puede recorrer el conector j si ha conseguido una llave de tipo $c[j]$ antes.

En cualquier punto del juego, el jugador está en un cuarto x y puede realizar dos tipos de acciones:

- recolectar la llave en el cuarto x, cuyo tipo es $r[x]$ (a menos que ya la haya recolectado antes),
- Recorrer el conector j, donde $u[j] = x$ o $v[j] = x$, si el jugador ha recolectado una llave de tipo $c[j]$ anteriormente. Note que el jugador **nunca** desecha una llave que ha recolectado.

El jugador **comienza** el juego en un cuarto s sin ninguna llave. Un cuarto t es **alcanzable** desde un cuarto s, si el jugador que comienza el juego en el cuarto s puede realizar alguna secuencia de acciones como las descritas anteriormente, y alcanzar el cuarto t.

Por cada cuarto i ($0 \leq i \leq n-1$), sea $p[i]$ el número de cuartos alcanzables desde el cuarto i. Timothy quisiera saber el conjunto de índices i que obtengan el mínimo valor de $p[i]$ para $0 \leq i \leq n-1$.

Detalles de Implementación

Usted debe implementar el siguiente procedimiento:

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

- r: un arreglo de longitud n. Para cada i ($0 \leq i \leq n-1$), la llave en el cuarto i es de tipo $r[i]$.
- u, v: dos arreglos de longitud m. Por cada j ($0 \leq j \leq m-1$), el conector j que conecta los cuartos $u[j]$ y $v[j]$.

- c: un arreglo de longitud m. Por cada j ($0 \leq j \leq m-1$), el tipo de llave necesaria para recorrer el conector j es $c[j]$.
- El procedimiento debe retornar un arreglo s de longitud n. Por cada $0 \leq i \leq n-1$, el valor de $s[i]$ debe ser 1 si por cada j tal que $0 \leq j \leq n-1$, $p[i] \leq p[j]$. De otra manera, el valor de $a[i]$ debe 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 comienza el juego en el cuarto 0, puede realizar la siguiente secuencia de acciones:

Cuarto Actual	Acción
0	Recolectar la llave de tipo 0
0	Recorrer el conector 0 al cuarto 1
1	Recolectar la llave de tipo 1
1	Recorrer el conector 2 al cuarto 2
2	Recorrer el conector 2 al cuarto 1
1	Recorrer el conector 3 al cuarto 3

Como consecuencia el cuarto 3 es alcanzable desde el cuarto 0. Del mismo modo, podemos construir secuencias para probar que todos los cuartos son alcanzables desde el cuarto 0, lo que implica que $p[0] = 4$. La tabla a continuación muestra los cuartos alcanzables comenzando desde cada cuarto:

Cuarto inicial i	Cuartos Alcanzables	$p[i]$
0	$[0, 1, 2, 3]$	4
1	$[1, 2]$	2
2	$[1, 2]$	2
3	$[1, 2, 3]$	3

El menor valor de $p[i]$ a través de todos los cuartos es 2, y esto se obtiene por $i = 1$ o $i = 2$. Entonces, el procedimiento debe 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 tabla a continuación muestra los cuartos alcanzables:

Cuarto inicial i	Cuartos alcanzables	$p[i]$
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 menor valor de $p[i]$ a través de todos los cuartos es 2, y esto se obtiene para $i \in \{1, 2, 4, 6\}$. Entonces, este procedimiento debe retornar $[0, 1, 1, 0, 1, 0, 1]$.

Ejemplo 3

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

La tabla a continuacióon muestra los cuartos alcanzables:

Cuarto inicial i	Cuartos alcanzables	$p[i]$
0	$[0, 1]$	2
1	$[0, 1]$	2
2	$[2]$	1

El menor valor de $p[i]$ a través de todos los cuartos es 1, y esto se obtiene cuando $i = 2$. Entonces, el procedimiento debe 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 $0 \leq j \leq m - 1$

- $0 \le c[j] \le n-1$ para todo $0 \le j \le m-1$

Subtareas

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

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

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

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

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