Árbol de Haya

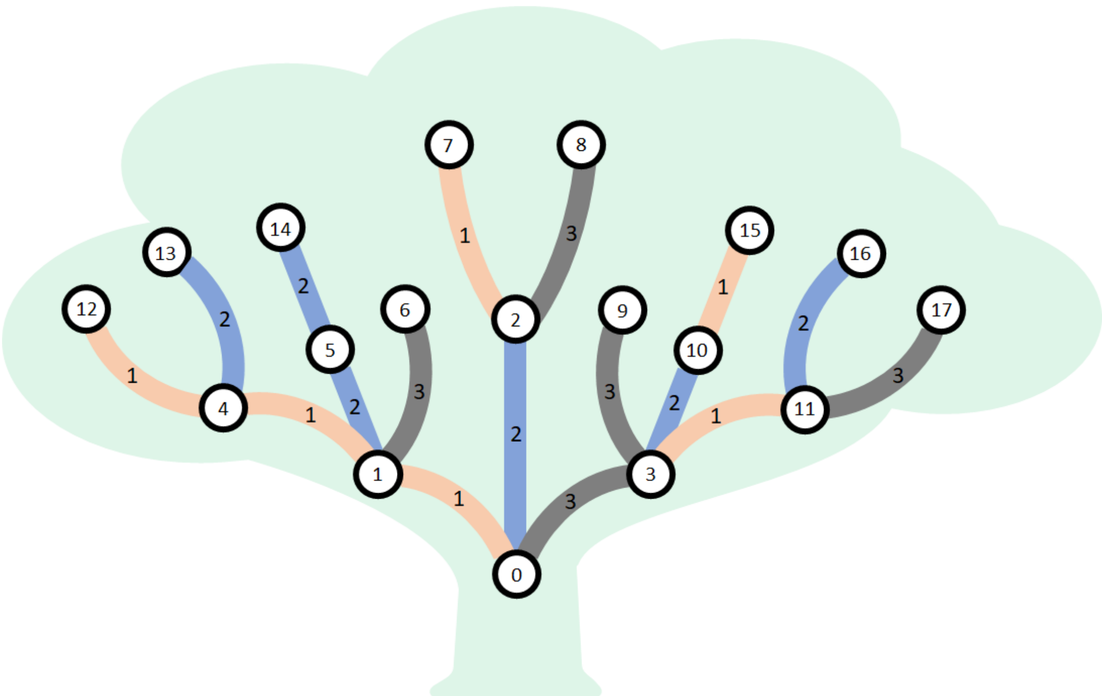
El Bosque de Vétyem es un bosque famoso con una variedad de árboles coloridos. Uno de los más antiguos y grandes árboles de Haya se llama Ős Vezér.

El árbol Ős Vezér puede ser representado como un conjunto de N **nodos** y $N-1$ **aristas**. Los nodos están numerados del 0 al $N-1$ y las aristas están numeradas del 1 al $N-1$. Cada arista conecta dos nodos distintos del árbol. Específicamente, la arista i ($1 \leq i < N$) conecta el nodo i con el nodo $P[i]$, donde $0 \leq P[i] < i$. El nodo $P[i]$ se lo denomina como **padre** del nodo i, y el nodo i se lo denomina como **hijo** del nodo $P[i]$.

Cada arista tiene un color. Hay M posibles colores de aristas numerados del 1 al M. El color de la arista i es $C[i]$. Distintas aristas pueden tener el mismo color.

Nótese que en las definiciones que anteceden, el caso $i=0$ no corresponde a una arista del árbol. Por conveniencia, decimos que $P[0]=-1$ y $C[0]=0$.

Por ejemplo, supongamos que Ős Vezér tiene $N=18$ nodos y $M=3$ posibles colores de aristas, con 17 aristas descritas por las conexiones $P=[-1,0,0,0,1,1,1,2,2,3,3,3,4,4,5,10,11,11]$ y colores $C=[0,1,2,3,1,2,3,1,3,3,2,1,1,2,2,1,2,3]$. El árbol se muestra en la siguiente figura:



Árpád es un talentoso guardabosques al que le gusta estudiar partes específicas del árbol llamadas **subárboles**. Para cada r tal que $0 \leq r < N$, el subárbol del nodo r es el conjunto $T(r)$ de nodos con las siguientes propiedades:

- El nodo r pertenece a $T(r)$.
- Si un nodo x pertenece a $T(r)$, todos los hijos de x también pertenecen a $T(r)$.
- Ningún otro nodo pertenece a $T(r)$.

El tamaño del conjunto $T(r)$ se denota por $|T(r)|$.

Árpád recientemente descubrió una propiedad complicada pero interesante para los subárboles. Su descubrimiento le requirió varios ensayos con lápiz y papel, y sospecha que tú también deberías hacer lo mismo para entenderlo. Además te enseñará múltiples ejemplos que puedes analizar a detalle.

Supón que tenemos un r fijo y una permutación $v_0, v_1, \ldots, v_{|T(r)|-1}$ de los nodos en el subárbol $T(r)$.

Para cada i tal que $1 \leq i < |T(r)|$, sea $f(i)$ el número de veces que el color $C[v_i]$ aparece en la siguiente secuencia de $i-1$ colores: $C[v_1], C[v_2], \ldots, C[v_{i-1}]$.

(Nótese que $f(1)$ siempre es 0 porque la secuencia de colores en su definición es vacía.)

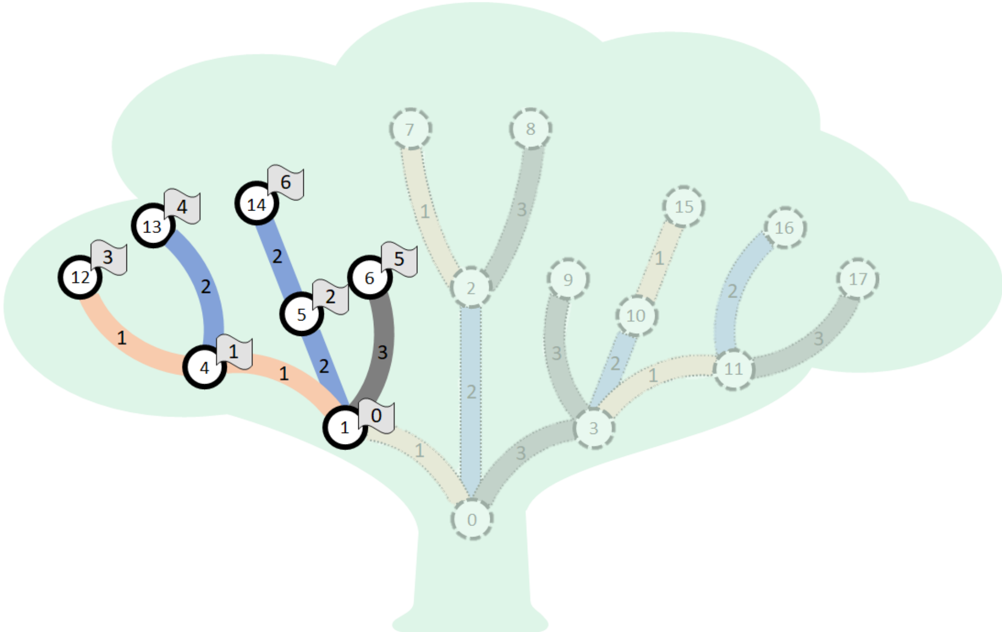
La permutación $v_0, v_1, \ldots, v_{|T(r)|-1}$ es una **permutación bonita** si y solo si todas las siguientes propiedades se cumplen:

- $v_0 = r$.
- Para cada i tal que $1 \leq i < |T(r)|$, el padre del nodo v_i es el nodo $v_{f(i)}$.

Para cada r tal que $0 \leq r < N$, el subárbol $T(r)$ es un **subárbol bonito** si y solo si existe una permutación bonita de los nodos que pertenecen a $T(r)$. Nótese que acorde a la definición cada subárbol que consiste de un único nodo es bonito.

Considera el ejemplo del árbol de arriba. Puede ser demostrado que los subárboles $T(0)$ y $T(3)$ de este árbol no son bonitos. El subárbol $T(14)$ es bonito, ya que consiste de un único nodo. A continuación demostraremos que el subárbol $T(1)$ también es bonito.

Considera la secuencia de enteros distintos $[v_0, v_1, v_2, v_3, v_4, v_5, v_6] = [1, 4, 5, 12, 13, 6, 14]$. Esta secuencia es una permutación de nodos que pertenecen a $T(1)$. La siguiente figura muestra esta permutación. Las etiquetas en los nodos son los índices en los cuales esos nodos aparecen en la permutación.



Ahora verificaremos que la esta es una *permutación bonita*.

- $v_0 = 1$.
- $f(1) = 0$ ya que $C[v_1] = C[4] = 1$ aparece 0 veces en la secuencia $[]$.
- Correspondientemente el padre de $[v_1]$ es v_0. Es decir, el padre de 4 es 1. (Formalmente, $P[4] = 1$.)
- $f(2) = 0$ ya que $C[v_2] = C[5] = 2$ aparece 0 veces en la secuencia $[1]$.
- Correspondientemente el padre de v_2 es v_0. Es decir, el padre de 5 es 1.
- $f(3) = 1$ ya que $C[v_3] = C[12] = 1$ aparece 1 vez en la secuencia $[1, 2]$.
- Correspondientemente el padre de v_3 es v_1. Es decir, el padre de 12 es 4.
- $f(4) = 1$ ya que $C[v_4] = C[13] = 2$ aparece 1 vez en la secuencia $[1, 2, 1]$.
- Correspondientemente el padre de v_4 es v_1. Es decir, el padre de 13 es 4.
- $f(5) = 0$ ya que $C[v_5] = C[6] = 3$ aparece 0 veces en la secuencia $[1, 2, 1, 2]$.
- Correspondientemente el padre de v_5 es v_0. Es decir, el padre de 6 es 1.
- $f(6) = 2$ ya que $C[v_6] = C[14] = 2$ aparece 2 veces en la secuencia $[1, 2, 1, 2, 3]$.
- Correspondientemente el padre de v_6 es v_2. Es decir, el padre de 14 es 5.

Ya que pudimos encontrar una *permutación bonita* de los nodos en $T(1)$, el subárbol $T(1)$ es un *subárbol bonito*.

Tu tarea es ayudar a Árpád a decidir si cada subárbol de Ős Vezér es bonito o no.

Detalles de Implementación

Debes implementar el siguiente método.

```
int[] beechtree(int N, int M, int[] P, int[] C)
```

- N: la cantidad de nodos del árbol.

- M: la cantidad de posibles colores de aristas.
- P, C: arreglos de tamaño N que describen las aristas del árbol.
- El método debe regresar un arreglo b de tamaño N. Para cada r tal que $0 \leq r < N$, $b[r]$ debe ser 1 si $T(r)$ es bonito, y 0 en caso contrario.
- Este método se llama exactamente una vez para cada caso de prueba.

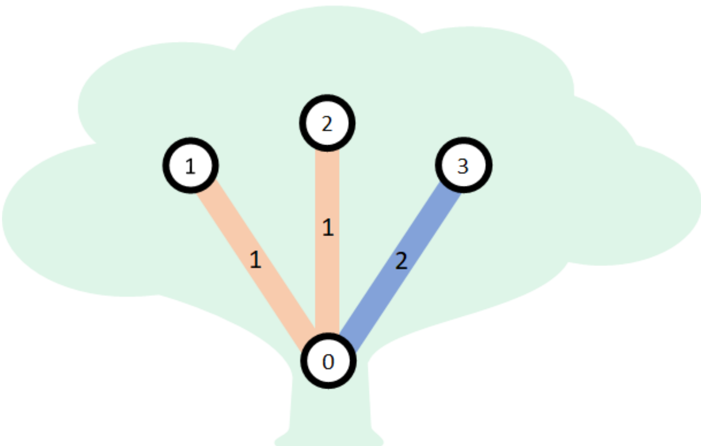
Ejemplos

Ejemplo 1

Considera la siguiente llamada:

```
beechtree(4, 2, [-1, 0, 0, 0], [0, 1, 1, 2])
```

El árbol se muestra en la siguiente figura:



$T(1)$, $T(2)$, y $T(3)$ cada uno consiste de un único nodo y por lo tanto son bonitos. $T(0)$ no es bonito. Por lo tanto, el método debe regresar $[0, 1, 1, 1]$.

Ejemplo 2

Considera la siguiente llamada:

```
beechtree(18, 3,
          [-1, 0, 0, 0, 1, 1, 1, 2, 2, 3, 3, 3, 4, 4, 5, 10, 11, 11],
          [0, 1, 2, 3, 1, 2, 3, 1, 3, 3, 2, 1, 1, 2, 2, 1, 2, 3])
```

Este ejemplo se muestra en la descripción del problema más arriba.

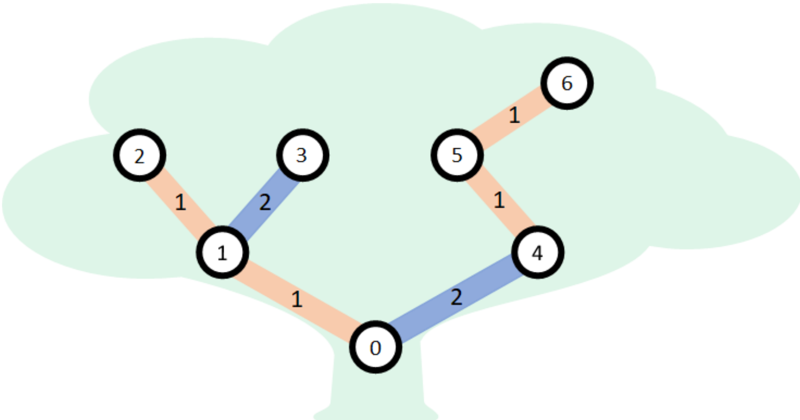
El método debe regresar $[0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$.

Ejemplo 3

Considera la siguiente llamada:

```
beechtree(7, 2, [-1, 0, 1, 1, 0, 4, 5], [0, 1, 1, 2, 2, 1, 1])
```

Este ejemplo se muestra en la siguiente figura.



$T(0)$ es el único subárbol que no es bonito. El método debe regresar $[0, 1, 1, 1, 1, 1, 1]$.

Restricciones

- $3 \le N \le 200\,000$
- $2 \le M \le 200\,000$
- $0 \le P[i] < v$ (para cada i tal que $1 \le i < N$)
- $1 \le C[i] \le M$ (para cada i tal que $1 \le i < N$)
- $P[0] = -1$ y $C[0] = 0$

Subtareas

1. (9 puntos) $N \le 8$ y $M \le 500$
2. (5 puntos) La arista i conecta al nodo i al nodo $i - 1$. Esto es, para cada i tal que $1 \le i < N$, $P[i] = i - 1$.
3. (9 puntos) Cada nodo distinto al nodo 0 está conectado al nodo 0, o está conectado a un nodo que está conectado al nodo 0. Esto es, para cada i tal que $1 \le i < N$, se cumple que $P[i] = 0$ o $P[P[i]] = 0$, pero no ambas.
4. (8 puntos) Para cada c tal que $1 \le c \le M$, hay a lo mucho dos aristas de color c.
5. (14 puntos) $N \le 200$ y $M \le 500$
6. (14 puntos) $N \le 2\,000$ y $M = 2$
7. (12 puntos) $N \le 2\,000$
8. (17 puntos) $M = 2$
9. (12 puntos) Sin restricciones adicionales.

Evaluador Local

El evaluador local lee la entrada en el siguiente formato:

- línea 1: $N \ M$
- línea 2: $P[0] \ P[1] \ \ldots \ P[N-1]$
- línea 3: $C[0] \ C[1] \ \ldots \ C[N-1]$

Sean $b[0], \ b[1], \ \ldots$ los elementos regresados por `beechtree`. El evaluador local imprime tu respuesta en una única línea, en el siguiente formato:

- línea 1: $b[0] \ b[1] \ \ldots$