Pohon Beech

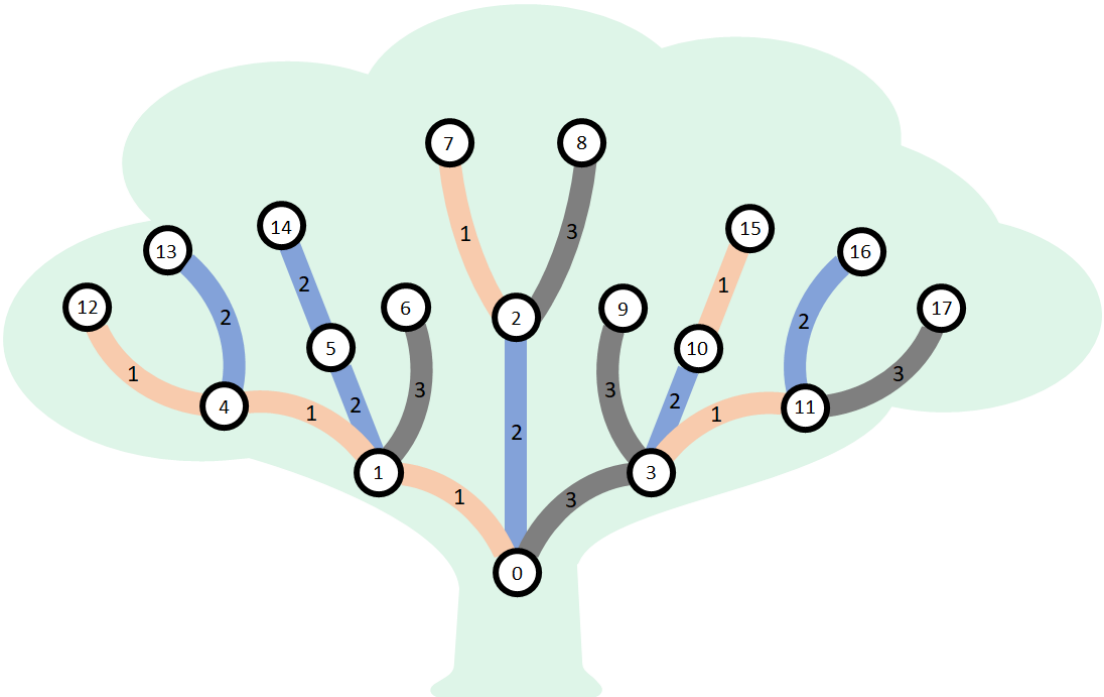
Vétyem Woods adalah hutan yang terkenal akan pohon-pohonnya yang berwarna. Salah satu pohon beech tertua dan tertingginya bernama Ős Vezér.

Pohon Ős Vezér bisa dimodelkan sebagai sebuah himpunan dengan N buah ***node*** dan $N-1$ buah ***edge***. *Node* dinomori dari 0 sampai $N-1$ dan *edge* dinomori dari 1 sampai $N-1$. Tiap *edge* menghubungkan dua buah *node* berbeda dari pohon. Lebih tepatnya, edge i ($1 \leq i < N$) menghubungkan *node* i dan *node* $P[i]$, dengan $0 \leq P[i] < i$. *Node* $P[i]$ disebut ***parent*** dari *node* i, dan node i disebut **anak** dari node $P[i]$.

Tiap *edge* mempunyai sebuah warna. Terdapat M kemungkinan warna yang dinomori dari 1 sampai M. Warna dari *edge* i adalah $C[i]$. *Edge* yang berbeda bisa mempunyai warna yang sama.

Perhatikan bahwa berdasarkan definisi di atas, kasus $i = 0$ tidak dipasangkan dengan *edge* pada pohon. Demi kenyamanan, kita anggap $P[0] = -1$ dan $C[0] = 0$.

Sebagai contoh, anggaplah Ős Vezér mempunyai $N = 18$ *node* dan $M = 3$ kemungkinan warna *edge*, dengan 17 *edge* yang diberikan dengan hubungan $P = [-1, 0, 0, 0, 1, 1, 1, 2, 2, 3, 3, 3, 4, 4, 5, 10, 11, 11]$ dan warna $C = [0, 1, 2, 3, 1, 2, 3, 1, 3, 3, 2, 1, 1, 2, 2, 1, 2, 3]$. Pohon tersebut diilustrasikan dengan gambar berikut:



Árpád adalah seorang pengawas hutan berbakat yang senang mempelajari bagian-bagian pohon tertentu yang dinamakan **subpohon**. Untuk setiap r sedemikian sehingga $0 \leq r < N$, subpohon dari *node* r adalah sebuah himpuan $T(r)$ dengan sifat-sifat berikut:

- Node r berada di dalam $T(r)$.
- Ketika suatu node x berada di dalam $T(r)$, semua anak-anak dari x juga berda di dalam $T(r)$.
- Tidak ada node lain yang berada di dalam $T(r)$.

Ukuran dari himpunan $T(r)$ dinotasikan dengan $|T(r)|$.

Árpád baru saja menemukan sifat rumit nan menarik dari sifat subpohon. Penemuan Árpád melibatkan banyak coret-coretan, dan ia mengharapkan Anda untuk melakukan hal yang sama untuk mengerti hal tersebut. Ia akan menunjukkan Anda beberapa contoh yang dapat Anda analisis secara rinci.

Misalkan kita memiliki r tetap dan sebuah permutasi $v_0, v_1, \ldots, v_{|T(r)|-1}$ dari $T(r)$.

Untuk setiap i sedemikian sehingga $1 \leq i < |T(r)|$, misalkan $f(i)$ adalah banyaknya kemunculan warna $C[v_i]$ pada barisan $C[v_1], C[v_2], \ldots, C[v_{i-1}]$.

(Perhatikan bahwa $f(1)$ selalu 0 karena barisan warna dalam definisinya adalah kosong.)

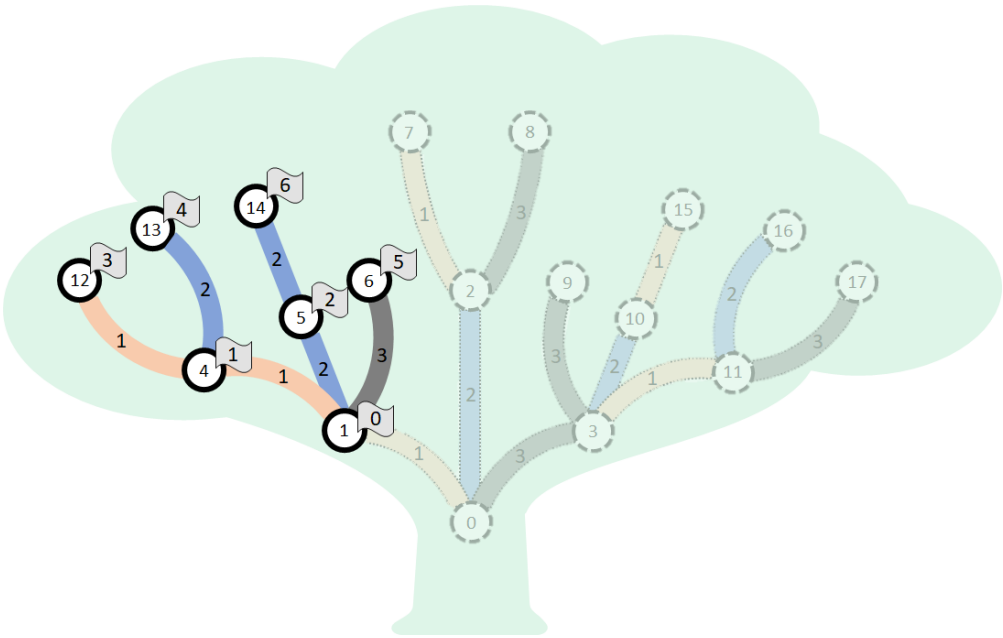
Permutasi $v_0, v_1, \ldots, v_{|T(r)|-1}$ adalah **permutasi cantik** jika dan hanya jika seluruh sefiat berikut berlaku:

- $v_0 = r$.
- Untuk setiap i sedemikian sehingga $1 \leq i < |T(r)|$, *parent* dari v_i adalah $v_{f(i)}$.

Untuk r mana pun sedemikian sehingga $0 \leq r < N$, subpohon $T(r)$ adalah **subpohon cantik** jika dan hanya jika terdapat sebuah permutasi cantik dari *node* dalam $T(r)$. Perhatikan bahwa menurut definisi setiap subpohon yang mengandung sebuah *node* dikatakan cantik.

Perhatikan contoh pohon di atas. Dapat ditunjukkan bahwa subpohon $T(0)$ dan $T(3)$ dari pohon ini tidaklah cantik. Subpohon $T(14)$ adalah cantik, karena dia hanya mengandung sebuah *node*. Di bawah ini, kita akan menunjukkan bahwa subpohon $T(1)$ jugalah cantik.

Perhatikan barisan bilangan bulat berbeda $[v_0, v_1, v_2, v_3, v_4, v_5, v_6] = [1, 4, 5, 12, 13, 6, 14]$. Barisan ini adalah permutasi dari *node* dalam $T(1)$. Gambar berikut menunjukkan permutasi ini. Label yang ditempel pada *node* adalah indeks ketika *node* tersebut muncul dalam permutasi.



Jelas bahwa barisan di atas merupakan sebuah permutasi dari *node* dalam $T(1)$. Kita akan menunjukkan bahwa $T(1)$ adalah *cantik*.

- $v_0 = 1$.
- $f(1) = 0$ karena $C[v_1] = C[4] = 1$ muncul 0 kali pada barisan $[]$.
  - Dengan demikian, *parent* dari v_1 adalah v_0. Dengan kata lain, *parent* dari 4 adalah 1.
- $f(2) = 0$ karena $C[v_2] = C[5] = 2$ muncul 0 kali pada barisan $[1]$.
  - Dengan demikian, *parent* dari v_2 adalah v_0. Dengan kata lain, *parent* dari 5 adalah 1.
- $f(3) = 1$ karena $C[v_3] = C[12] = 1$ muncul 1 kali pada barisan $[1, 2]$.
  - Dengan demikian, *parent* dari v_3 adalah v_1. Dengan kata lain, *parent* dari 12 adalah 4.
- $f(4) = 1$ karena $C[v_4] = C[13] = 2$ muncul 1 kali pada barisan $[1, 2, 1]$.
  - Dengan demikian *parent* dari v_4 adalah v_1. Dengan kata lain, *parent* dari 13 adalah 4.
- $f(5) = 0$ karena $C[v_5] = C[6] = 3$ muncul 0 kali pada barisan $[1, 2, 1, 2]$.
  - Dengan demikian, *parent* dari v_5 adalah v_0. Dengan kata lain, *parent* dari 6 adalah 1.
- $f(6) = 2$ karena $C[v_6] = C[14] = 2$ muncul 2 kali pada barisan $[1, 2, 1, 2, 3]$.
  - Dengan demikian, *parent* dari v_6 adalah v_2. Dengan kata lain, *parent* dari 14 adalah 5.

Karena kita dapat menemukan permutasi cantik untuk *node-node* dalam $T(1)$, subpohon $T(1)$ memanglah cantik.

Tugas Anda adalah membantu Árpád untuk menentukan untuk setiap subpohon dari Ős Vezér apakah subpohon tersebut adalah cantik.

Detail Implementasi

Anda harus mengimplementasikan prosedur berikut.

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

- N: banyaknya *node* dalam pohon.
- M: banyaknya kemungkinan warna *edge*.
- P, C: dua *array* sepanjang N yang mendeskripsikan *edge* dari pohon.
- Prosedur ini harus mengembalikan sebuah *array* b sepanjang N. Untuk setiap r sedemikian sehingga $0 \leq r < N$, $b[r]$ harus bernilai 1 jika $T(r)$ adalah cantik, dan 0 bila sebaliknya.
- Prosedur ini dipanggil tepat sekali untuk setiap kasus uji.

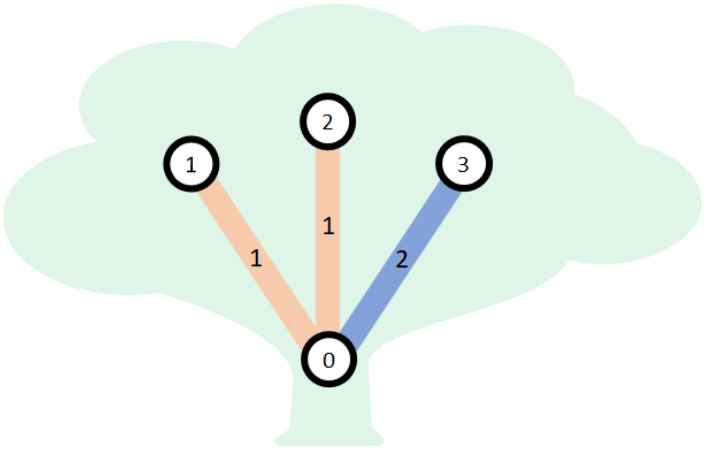
Contoh

Contoh 1

Perhatikan pemanggilan berikut:

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

Pohon ini ditampilkan dengan gambar berikut:



$T(1)$, $T(2)$, dan $T(3)$ masing-masing mengandung sebuah *node* sehingga semuanya adalah cantik. $T(0)$ tidaklah cantik. Maka dari itu, prosedur harus mengembalikan $[0, 1, 1, 1]$.

Contoh 2

Perhatikan pemanggilan berikut:

```
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])
```

Contoh ini diilustrasikan pada deskripsi soal di atas.

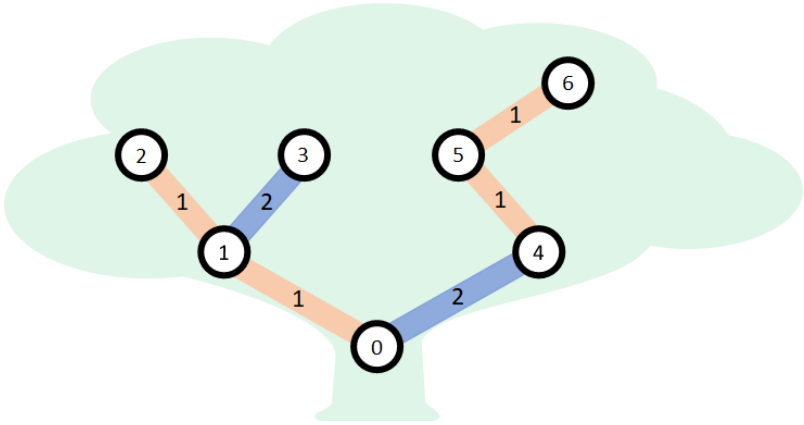
Prosedur ini harus mengembalikan $[0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$.

Contoh 3

Perhatikan pemanggilan berikut:

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

Contoh ini diilustrasikan dengan gambar berikut.



$T(0)$ adalah satu-satunya subpohon yang tidak cantik. Prosedur ini harus mengembalikan $[0, 1, 1, 1, 1, 1, 1]$.

Batasan

- $3 \leq N \leq 200\,000$
- $2 \leq M \leq 200\,000$
- $0 \leq P[i] < v$ (untuk setiap i sedemikian sehingga $1 \leq i < N$)
- $1 \leq C[i] \leq M$ (untuk setiap i sedemikian sehingga $1 \leq i < N$)
- $P[0] = -1$ dan $C[0] = 0$

Subsoal

1. (9 poin) $N \leq 8$ dan $M \leq 500$
2. (5 poin) *Edge* i menghubungkan *node* i dengan *node* $i - 1$. Dengan kata lain, untuk setiap i sedemikian sehingga $1 \leq i < N$, $P[i] = i - 1$.
3. (9 poin) Tiap *node* selain *node* 0 menghubungkan *node* 0, atau terhubung dengan sebuah *node* yang terhubung dengan *node* 0. Dengan kata lain, untuk setiap v sedemikian sehingga $1 \leq v < N$, $P[v] = 0$ atau $P[P[v]] = 0$.
4. (8 poin) Untuk setiap c sedemikian sehingga $1 \leq c \leq M$, terdapat paling banyak dua *edge* dengan warna c.
5. (14 poin) $N \leq 200$ dan $M \leq 500$
6. (14 poin) $N \leq 2\,000$ dan $M = 2$
7. (12 poin) $N \leq 2\,000$
8. (17 poin) $M = 2$
9. (12 poin) Tidak ada batasan tambahan.

Contoh *Grader*

Contoh *grader* membaca masukan dengan format berikut:

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

Misalkan $b[0],\ b[1],\ \ldots$ menyatakan elemen-elemen dari *array* yang dikembalikan oleh `beechtree`.
Contoh *grader* mencetak jawaban Anda dalam satu baris, dengan format berikut:

- baris 1: $b[0]\ b[1]\ \ldots$