Манкунианец и цветное дерево

После напряженной недели на работе, жители Манчестера и Ливерпуля решили пойти в поход на выходные. Когда они проходили по лесу, они наткнулись на уникальное дерево, состоящее из *n* вершин. Вершины пронумерованы числами от 1 до *n*.

Каждой вершине дерева поставлен в соответствие цвет (из c возможных цветов). Чтобы побороть скуку, они решили проверить вместе свои навыки рассуждения. Корнем дерева является вершина 1. Для каждой вершины они решили найти ближайшего предка, цвет которого совпадает с цветом вершины.

**Вход.** Первая строка содержит два целых числа *n* и *c* (1 ≤ *n*, *c* ≤ 105) – количество вершин в дереве и количество возможных цветов.

Вторая строка содержит *n* – 1 число. *i*-ое число указывает на отца *i* + 1 - ой вершины.

Третья строка содержит *n* целых чисел, задающих цвета вершин. Значения цветов лежат в промежутке от 1 до *c* включительно.

**Выход.**  В одной строке выведите *n* чисел. *i*-ым числом является вершина, являющаяся ближайшим предком *i*-ой вершины, имеющей такой же цвет. Если такого предка для вершины не существует, то вывести -1.

|  |  |
| --- | --- |
| **Пример входа** | **Пример выхода** |
| 5 4  1 1 3 3  1 4 2 1 2 | -1 -1 -1 1 3 |

РЕШЕНИЕ

**графы – поиск в глубину**

Анализ алгоритма

Рассмотрим более простой вариант задачи. Пусть все вершины дерева покрашены в один цвет. Заведем стек, который сначала пустой. Запустим поиск в глубину из корня дерева. При входе в вершину *v* кладем в стек значение *v*, а при завершении обработки вершины *v* удаляем верхушку стека (это как раз будет вершина *v*). Когда поиск в глубину завершится, стек окажется пустым.

**Вопрос**: что будет на вершине стека при входе в вершину *v* и еще до того как мы положим *v* в стек?

Теперь перейдем к решению нашей задачи. Заведем *c* стеков, по одному для каждого цвета (например, вектор стеков). Изначально все стеки пустые. Запустим поиск в глубину из корня – вершины 1. Обработка вершины *v* цвета *color* состоит из следующих шагов:

* Если стек s[*color*] не пуст, то на его вершине находится номер вершины, являющейся ближайшим предком *v*, имеющей такой же цвет как и *v*. Если стек пуст, то требуемой вершины не существует, ответом для вершины *v* будет -1.
* Заносим вершину *v* в стек s[*color*].
* Запускаем поиск в глубину со всех сыновей вершины *v*.
* Удаляем вершину из стека s[*color*].

Когда мы находимся при поиске в глубину в вершине *v*, в стеках хранится информация о цветах всех вершин, расположенных на единственном пути от корня до *v*. То есть стек s[*color*] содержит номера вершин на пути от корня до *v*, которые имеют цвет *color*. При этом вершины в стек заносятся в порядке их посещения поиском в глубину.

**Пример**

Когда поиск в глубину дойдет до вершины 5, из *c* = 4 стеков два будут пустыми (соответствующие цветам 3 и 4). Стек номер 1 (первый цвет) содержит вершину 1, Стек номер 2 (второй цвет) содержит вершину 3. Вершина номер 5 имеет цвет 2, следовательно ближайшим предком с тем же цветом будет вершина, находящаяся на вершине стека 2. Таким предком для вершины 5 будет вершина 3.

