열쇠

건축가 티모시는 새로운 탈출 게임을 설계했다. 이 게임에는 n 개의 방이 있다. 방들은 0번부터 $n-1$번까지 번호가 붙어 있다. 처음에 각 방에는 정확히 하나의 열쇠가 있다. 각 열쇠는 하나의 타입에 속하는데, 타입들은 0번부터 $n-1$번까지 번호가 붙어 있다. 방들 중 $i (0 \le i \le n-1)$번 방에 있는 열쇠의 타입은 $r[i]$이다. 여러개의 방이 같은 타입의 열쇠를 가지고 있을 수도 있다. 즉 $r[i]$들 중에 같은 값들이 있을 수 있다.

방들을 잇는 m개의 **양방향** 복도들이 있다. 복도들은 0번부터 $m-1$번까지 번호가 붙어 있다. 복도들 중 $j (0 \le j \le m-1)$번 복도는 $u[j]$번 방과 $v[j]$번 방을 연결한다 ($u[j] \neq v[j]$). 한 쌍의 방을 연결하는 복도가 여러개일 수 있다.

이 게임은 참가자 한 명이 플레이한다. 참가자는 방들을 다니며 열쇠를 모으며 게임을 플레이한다. 참가자가 $u[j]$번 방에서 $v[j]$번 방으로 (혹은 그 반대로) j번 복도를 이용해서 이동하는 것을 j번 복도를 **사용한다**고 부른다. 참가자가 j번 복도를 사용하기 위해서는 그 전에 $c[j]$ 타입의 열쇠를 얻었어야 한다.

게임 하는 도중 x번 방에서 참가자가 할 수 있는 행동은 아래 두가지이다. (둘 다 할 수도 있다.)

- (동일한 타입의 열쇠를 이미 가지고 있지 않다면) x번 방의 열쇠를 얻는다. 이때 열쇠의 타입은 $r[x]$이다.
- $u[j] = x$ 혹은 $v[j] = x$인 복도를 사용해서 방을 떠난다. 단, 이 때 참가자는 타입 $c[j]$인 열쇠를 가지고 있어야 한다. 참가자가 열쇠를 버리는 경우는 없다.

참가자는 열쇠 없이 어떤 s번 방에서 **시작**한다. s번 방에서 시작했을 때 위의 행동을 임의로 반복해 t번 방에 갈수 있는 방법이 존재한다면 t번 방은 **도달가능**하다고 부른다.

각각의 $i (0 \le i \le n-1)$번 방에 대해서, i번 방에서 시작했을 때 도달가능한 방들의 개수를 $p[i]$라고 하자. 티모시는 $p[i]$가 최소가 되는 방번호 $i (0 \le i \le n-1)$들의 집합을 찾고 싶다.

Implementation Details

다음 함수를 구현해야 한다.

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

- r: 길이 n인 배열. 각 $i (0 \le i \le n-1)$에 대해, i번 방에 있는 열쇠의 타입은 $r[i]$이다.
- u, v: 길이 m인 배열들. 각 $j (0 \le j \le m-1)$에 대해, j번 복도는 $u[j]$번과 $v[j]$번 방을 연결한다.
- c: 길이 m인 배열. 각 $j (0 \le j \le m-1)$에 대해, j번 복도를 사용하기 위해 필요한 열쇠의 타입은 $c[j]$이다.
- 이 함수는 길이 n인 배열 a를 리턴해야 한다. 각 $i (0 \le i \le n-1)$에 대해, $a[i]$의 값은 $p[i]$가 다른 모든 $p[j]$보다 작거나 같은 경우 1, 그렇지 않은 경우 0이라야 한다.

Examples

Example 1

다음 호출을 보자.

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

참가자가 0번 방에서 시작하는 경우, 다음과 같은 과정을 통해서 3번 방에 갈 수 있다.

현재 방	행동
0	타입 0의 열쇠를 얻음
0	0번 복도를 사용해 1번 방으로 이동
1	타입 1의 열쇠를 얻음
1	2번 복도를 사용해 2번 방으로 이동
2	2번 복도를 사용해 1번 방으로 이동
1	3번 복도를 사용해 3번 방으로 이동

따라서, 3번 방은 0번 방에서 시작했을 때 도달가능하다. 비슷한 방법으로 다른 모든 방으로 가는 방법을 만들 수 있으므로 $p[0] = 4$이다. 아래 표는 각 방에서 시작했을 때 도달가능한 방들의 리스트를 보여 준다.

시작 방 번호 i	도달가능한 방들	$p[i]$
0	$[0, 1, 2, 3]$	4
1	$[1, 2]$	2
2	$[1, 2]$	2
3	$[1, 2, 3]$	3

모든 $p[i]$들 중 가장 작은 값은 2이며, $i = 1$ 혹은 $i = 2$인 경우들이다. 따라서 이 함수는 $[0, 1, 1, 0]$을 리턴해야 한다.

Example 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])
```

아래 표는 도달 가능한 방들을 보여 준다.

시작 방 번호 i	도달가능한 방들	$p[i]$
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

모든 $p[i]$들 중 가장 작은 값은 2이며, $i \in \{1, 2, 4, 6\}$인 경우들이다. 따라서 이 함수는 $[0, 1, 1, 0, 1, 0, 1]$을 리턴해야 한다.

Example 3

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

아래 표는 도달 가능한 방들을 보여 준다.

시작 방 번호 i	도달가능한 방들	$p[i]$
0	$[0, 1]$	2
1	$[0, 1]$	2
2	$[2]$	1

모든 $p[i]$들 중 가장 작은 값은 1이며, $i = 2$인 경우이다. 따라서 이 함수는 $[0, 0, 1]$을 리턴해야 한다.

Constraints

- $2 \le n \le 300\,000$
- $1 \le m \le 300\,000$
- $0 \le r[i] \le n - 1$ (모든 $0 \le i \le n - 1$)
- $0 \le u[j], v[j] \le n - 1$이고 $u[j] \neq v[j]$ (모든 $0 \le j \le m - 1$)
- $0 \le c[j] \le n - 1$ (모든 $0 \le j \le m - 1$)

Subtasks

1. (9 points) $c[j] = 0$ (모든 $0 \le j \le m - 1$). 또, $n, m \le 200$
2. (11 points) $n, m \le 200$

3. (17 points) $n, m \leq 2000$
4. (30 points) $c[j] \leq 29$ (모든 $0 \leq j \leq m-1$)이고 $r[i] \leq 29$ (모든 $0 \leq i \leq n-1$)
5. (33 points) 추가적인 제한이 없음

Sample Grader

Sample Grader 의 입력 양식은 아래와 같다.

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

Sample Grader는 `find_reachable`의 리턴 값을 다음과 같이 출력한다.

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