**[본 문제는 정확성과 효율성 테스트 각각 점수가 있는 문제입니다.]**

업무용 소프트웨어를 개발하는 니니즈웍스의 인턴인 앙몬드는 명령어 기반으로 표의 행을 선택, 삭제, 복구하는 프로그램을 작성하는 과제를 맡았습니다. 세부 요구 사항은 다음과 같습니다



위 그림에서 파란색으로 칠해진 칸은 현재 **선택된 행**을 나타냅니다. 단, 한 번에 한 행만 선택할 수 있으며, 표의 범위(0행 ~ 마지막 행)를 벗어날 수 없습니다. 이때, 다음과 같은 명령어를 이용하여 표를 편집합니다.

* "U X": 현재 선택된 행에서 X칸 위에 있는 행을 선택합니다.
* "D X": 현재 선택된 행에서 X칸 아래에 있는 행을 선택합니다.
* "C" : 현재 선택된 행을 삭제한 후, 바로 아래 행을 선택합니다. 단, 삭제된 행이 가장 마지막 행인 경우 바로 윗 행을 선택합니다.
* "Z" : 가장 최근에 삭제된 행을 원래대로 복구합니다. **단, 현재 선택된 행은 바뀌지 않습니다.**

예를 들어 위 표에서 "D 2"를 수행할 경우 아래 그림의 왼쪽처럼 4행이 선택되며, "C"를 수행하면 선택된 행을 삭제하고, 바로 아래 행이었던 "네오"가 적힌 행을 선택합니다(4행이 삭제되면서 아래 있던 행들이 하나씩 밀려 올라오고, 수정된 표에서 다시 4행을 선택하는 것과 동일합니다).



다음으로 "U 3"을 수행한 다음 "C"를 수행한 후의 표 상태는 아래 그림과 같습니다.



다음으로 "D 4"를 수행한 다음 "C"를 수행한 후의 표 상태는 아래 그림과 같습니다. 5행이 표의 마지막 행 이므로, 이 경우 바로 윗 행을 선택하는 점에 주의합니다.



다음으로 "U 2"를 수행하면 현재 선택된 행은 2행이 됩니다.



위 상태에서 "Z"를 수행할 경우 가장 최근에 제거된 "라이언"이 적힌 행이 원래대로 복구됩니다.



다시한번 "Z"를 수행하면 그 다음으로 최근에 제거된 "콘"이 적힌 행이 원래대로 복구됩니다. 이때, 현재 선택된 행은 바뀌지 않는 점에 주의하세요.  


이때, 최종 표의 상태와 처음 주어진 표의 상태를 비교하여 삭제되지 않은 행은 "O", 삭제된 행은 "X"로 표시하면 다음과 같습니다.



처음 표의 행 개수를 나타내는 정수 n, 처음에 선택된 행의 위치를 나타내는 정수 k, 수행한 명령어들이 담긴 문자열 배열 cmd가 매개변수로 주어질 때, 모든 명령어를 수행한 후 표의 상태와 처음 주어진 표의 상태를 비교하여 삭제되지 않은 행은 O, 삭제된 행은 X로 표시하여 문자열 형태로 return 하도록 solution 함수를 완성해주세요.

제한사항

* 5 ≤ n ≤ 1,000,000
* 0 ≤ k < n
* 1 ≤ cmd의 원소 개수 ≤ 200,000
  + cmd의 각 원소는 "U X", "D X", "C", "Z" 중 하나입니다.
  + X는 1 이상 300,000 이하인 자연수이며 0으로 시작하지 않습니다.
  + X가 나타내는 자연수에 ',' 는 주어지지 않습니다. 예를 들어 123,456의 경우 123456으로 주어집니다.
  + cmd에 등장하는 모든 X들의 값을 합친 결과가 1,000,000 이하인 경우만 입력으로 주어집니다.
  + 표의 모든 행을 제거하여, 행이 하나도 남지 않는 경우는 입력으로 주어지지 않습니다.
  + 본문에서 각 행이 제거되고 복구되는 과정을 보다 자연스럽게 보이기 위해 "이름" 열을 사용하였으나, "이름"열의 내용이 실제 문제를 푸는 과정에 필요하지는 않습니다. "이름"열에는 서로 다른 이름들이 중복없이 채워져 있다고 가정하고 문제를 해결해 주세요.
* 표의 범위를 벗어나는 이동은 입력으로 주어지지 않습니다.
* 원래대로 복구할 행이 없을 때(즉, 삭제된 행이 없을 때) "Z"가 명령어로 주어지는 경우는 없습니다.
* 정답은 표의 0행부터 n - 1행까지에 해당되는 O, X를 순서대로 이어붙인 문자열 형태로 return 해주세요.

정확성 테스트 케이스 제한 사항

* 5 ≤ n ≤ 1,000
* 1 ≤ cmd의 원소 개수 ≤ 1,000

효율성 테스트 케이스 제한 사항

* 주어진 조건 외 추가 제한사항 없습니다.

입출력 예

| **n** | **k** | **cmd** | **result** |
| --- | --- | --- | --- |
| 8 | 2 | ["D 2","C","U 3","C","D 4","C","U 2","Z","Z"] | "OOOOXOOO" |
| 8 | 2 | ["D 2","C","U 3","C","D 4","C","U 2","Z","Z","U 1","C"] | "OOXOXOOO" |

입출력 예 설명

**입출력 예 #1**

문제의 예시와 같습니다.

**입출력 예 #2**

다음은 9번째 명령어까지 수행한 후의 표 상태이며, 이는 입출력 예 #1과 같습니다.



10번째 명령어 "U 1"을 수행하면 "어피치"가 적힌 2행이 선택되며, 마지막 명령어 "C"를 수행하면 선택된 행을 삭제하고, 바로 아래 행이었던 "제이지"가 적힌 행을 선택합니다.



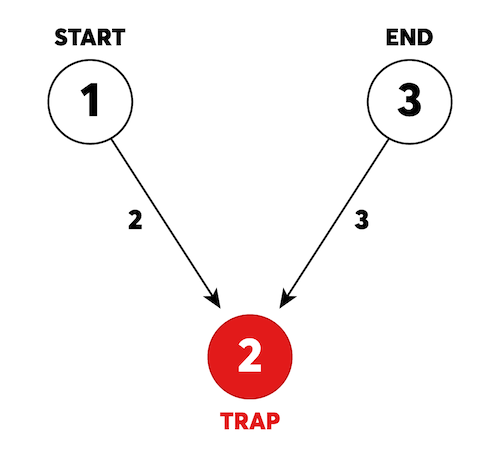
따라서 처음 주어진 표의 상태와 최종 표의 상태를 비교하면 다음과 같습니다.



제한시간 안내

* 정확성 테스트 : 10초
* 효율성 테스트 : 언어별로 작성된 정답 코드의 실행 시간의 적정 배수

신규 게임 ‘카카오 미로 탈출’이 출시되어, 라이언이 베타테스터로 참가했습니다.



위 예시 그림은 카카오 미로 탈출의 초기 상태를 나타냅니다. 1번부터 3번까지 번호가 붙어있는 3개의 방이 있고, 방과 방 사이를 연결하는 길에는 이동하는데 걸리는 시간이 표시되어 있습니다. 길은 화살표가 가리키는 방향으로만 이동할 수 있습니다. 미로에는 함정이 존재하며, 함정에 걸리면 함정과 연결된 모든 화살표의 방향이 바뀝니다.  
출발지점인 1번 방에서 탈출이 가능한 3번 방까지 이동해야 합니다. 탈출하는데 걸리는 최소 시간을 구하려고 합니다.

* 그림의 원은 방을 나타내며 원 안의 숫자는 방 번호를 나타냅니다.
  + 방이 n개일 때, 방 번호는 1부터 n까지 사용됩니다.
* 화살표에 표시된 숫자는 방과 방 사이를 이동할 때 걸리는 시간을 나타냅니다.
  + 화살표가 가리키고 있는 방향으로만 이동이 가능합니다. 즉, 위 그림에서 2번 방에서 1번 방으로는 이동할 수 없습니다.
* 그림에 표시된 빨간색 방인 2번 방은 함정입니다.
  + 함정 방으로 이동하는 순간, 이동한 함정 방과 연결되어있는 모든 길의 방향이 반대가 됩니다.
  + 위 그림 1번 방에서 2번 방으로 이동하는 순간 1에서 2로 이동할 수 있던 길은 2에서 1로 이동할 수 있는 길로 바뀌고, 3에서 2로 이동할 수 있던 길은 2에서 3으로 이동할 수 있는 길로 바뀝니다.
  + 똑같은 함정 방을 두 번째 방문하게 되면 원래 방향의 길로 돌아옵니다. 즉, 여러 번 방문하여 계속 길의 방향을 반대로 뒤집을 수 있습니다.
* 미로를 탈출하는데 필요한 최단 시간은 다음과 같습니다.
  + 1→2: 2번 방으로 이동합니다. 이동 시간은 2입니다.
  + 함정 발동: 2번 방과 연결된 모든 길의 방향이 반대가 됩니다.
  + 2→3: 3번 방으로 이동합니다. 이동 시간은 3입니다.
  + 탈출에 성공했습니다. 총 이동시간은 5입니다.

방의 개수를 나타내는 정수 n, 출발 방의 번호 start, 도착 방의 번호 end, 통로와 이동시간을 나타내는 2차원 정수 배열 roads, 함정 방의 번호를 담은 정수 배열 traps이 매개변수로 주어질 때, 미로를 탈출하는데 필요한 최단 시간을 return 하도록 solution 함수를 완성해주세요.

제한사항

* 1 ≤ n ≤ 1,000
* 1 ≤ start ≤ n
* 1 ≤ end ≤ n
* 1 ≤ roads의 행 길이 ≤ 3,000
* roads의 행은 [P, Q, S]로 이루어져 있습니다.
  + P에서 Q로 갈 수 있는 길이 있으며, 길을 따라 이동하는데 S만큼 시간이 걸립니다.
  + 1 ≤ P ≤ n
  + 1 ≤ Q ≤ n
  + P ≠ Q
  + 1 ≤ S ≤ 3,000
  + 서로 다른 두 방 사이에 여러 개의 길이 존재할 수도 있습니다.
* 0 ≤ traps의 길이 ≤ 10
  + 1 ≤ traps의 원소 ≤ n
  + 시작 방과 도착 방은 함정이 아닙니다.
* 항상 미로를 탈출할 수 있는 경우만 주어집니다.

입출력 예

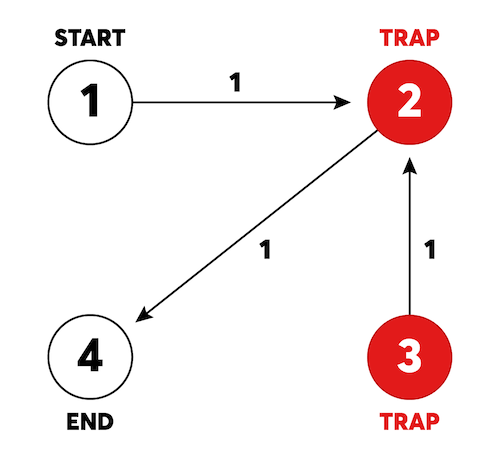
| **n** | **start** | **end** | **roads** | **traps** | **result** |
| --- | --- | --- | --- | --- | --- |
| 3 | 1 | 3 | [[1, 2, 2], [3, 2, 3]] | [2] | 5 |
| 4 | 1 | 4 | [[1, 2, 1], [3, 2, 1], [2, 4, 1]] | [2, 3] | 4 |

입출력 예 설명

**입출력 예 #1**

문제 예시와 같습니다.

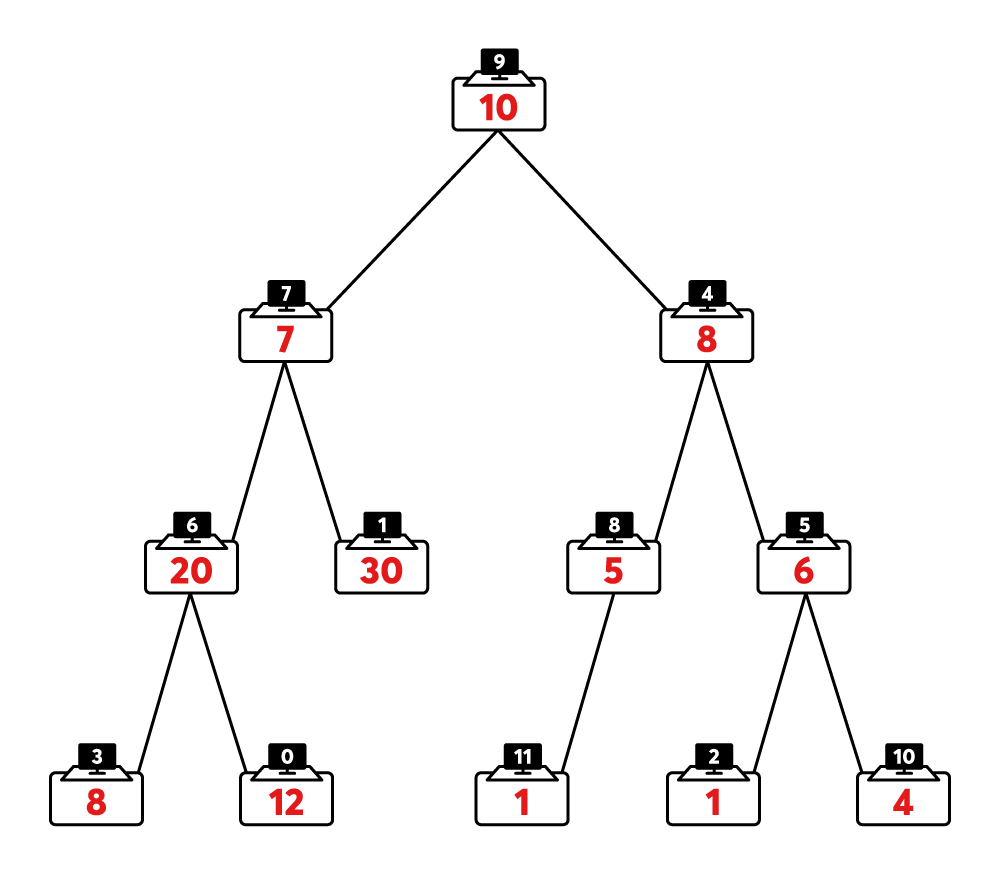
**입출력 예 #2**



1 → 2 → 3 → 2 → 4 순서로 이동하면 됩니다. 총 이동시간은 4입니다.

**[본 문제는 정확성과 효율성 테스트 각각 점수가 있는 문제입니다.]**

카카오 인턴을 선발하는 코딩 테스트 시험장이 하나의 이진 트리[1](https://programmers.co.kr/tryouts/23568/challenges#fn1) 형태로 연결되어 있습니다. 아래 그림은 12개의 시험장이 연결된 예시입니다.



1. 하나의 노드는 하나의 시험장을 나타냅니다.
2. 검은 바탕의 흰 숫자는 해당 시험장의 고유 번호(ID)를 나타냅니다.

2-1. 시험장이 n개 있다면, 시험장의 고유 번호는 0부터 n-1까지 부여됩니다.

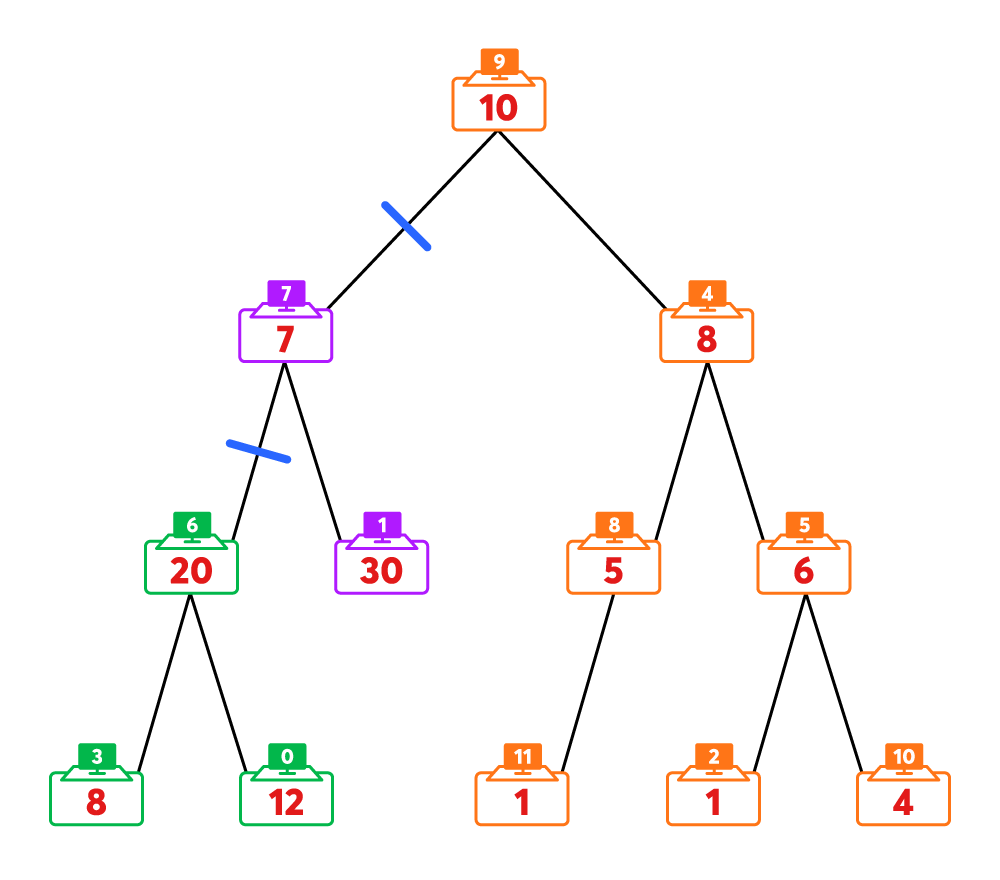
1. 노드 안의 빨간 숫자는, 해당 시험장의 응시자 수를 나타냅니다.

3-1. 위의 그림에서, 9번 시험장에는 10명, 4번 시험장에는 8명, 6번 시험장에는 20명의 응시자가 시험을 볼 예정입니다.

1. 노드 사이의 간선은 해당 시험장이 연결되어 있음을 의미합니다.

4-1. 위의 그림에서, 9번 시험장은 7번 시험장과, 7번 시험장은 6번 시험장과 연결되어 있습니다.

코딩 테스트를 총괄하는 무지는 안정적인 시험을 위해, 시험장에서 오는 트래픽을 k개의 그룹으로 나누어 각 그룹별 서버로 분산시키기로 하였습니다. 시험장 사이를 연결한 간선들 중 k-1개를 끊어서 시험장을 k 개의 그룹으로 나눌 계획입니다. 이때, 그룹별 최대 트래픽을 최소화하기 위하여 **가장 큰 그룹의 인원을 최소화시켜야 합니다.**



위의 그림에서 7번과 6번 시험장을 잇는 간선을 끊고, 9번과 7번 시험장을 잇는 간선을 끊는다면, 전체 시험장은 3개의 그룹으로 나누어집니다.

* 주황색 노드로 표시된 A그룹의 인원은 35명(10+8+5+6+1+1+4)
* 보라색 노드로 표시된 B그룹의 인원은 37명(7+30)
* 녹색 노드로 표시된 C그룹의 인원은 40명(20+8+12)

즉, 인원이 가장 많은 그룹은 40명입니다. 다른 어떤 방법으로 시험장을 3개의 그룹으로 나눈다고 해도, 인원이 가장 많은 그룹의 인원이 40명 미만이 되도록 나눌 수는 없습니다.

나눌 그룹의 수를 나타내는 정수 k, 각 시험장의 응시자 수를 나타내는 1차원 정수 배열 num, 시험장의 연결 상태를 나타내는 2차원 정수 배열 links가 매개변수로 주어집니다. 인원이 가장 많은 그룹의 인원이 최소화되도록 k개의 그룹으로 나누었을 때, 최소화된 최대 그룹의 인원을 return 하도록 solution 함수를 완성해주세요.

제한사항

* 1 ≤ k ≤ 10,000
* k ≤ num의 길이 ≤ 10,000
  + num[i]에는 i번 시험장의 응시자 수가 담겨있습니다.
  + 1 ≤ num의 원소 ≤ 10,000
* links의 길이 = num의 길이
  + links의 i번째 행은 i번 노드(시험장)의 [왼쪽 자식 노드 번호, 오른쪽 자식 노드 번호]입니다.
    - 해당 위치에 자식 노드가 없는 경우 -1이 담겨있습니다.
    - 잘못된 노드 번호나, 하나의 이진 트리 구조가 아닌 입력은 주어지지 않습니다.

정확성 테스트 케이스 제한 사항

* 1 ≤ k ≤ 20
* k ≤ num의 길이 ≤ 20

효율성 테스트 케이스 제한 사항

* 주어진 조건 외 추가 제한사항 없습니다.

입출력 예

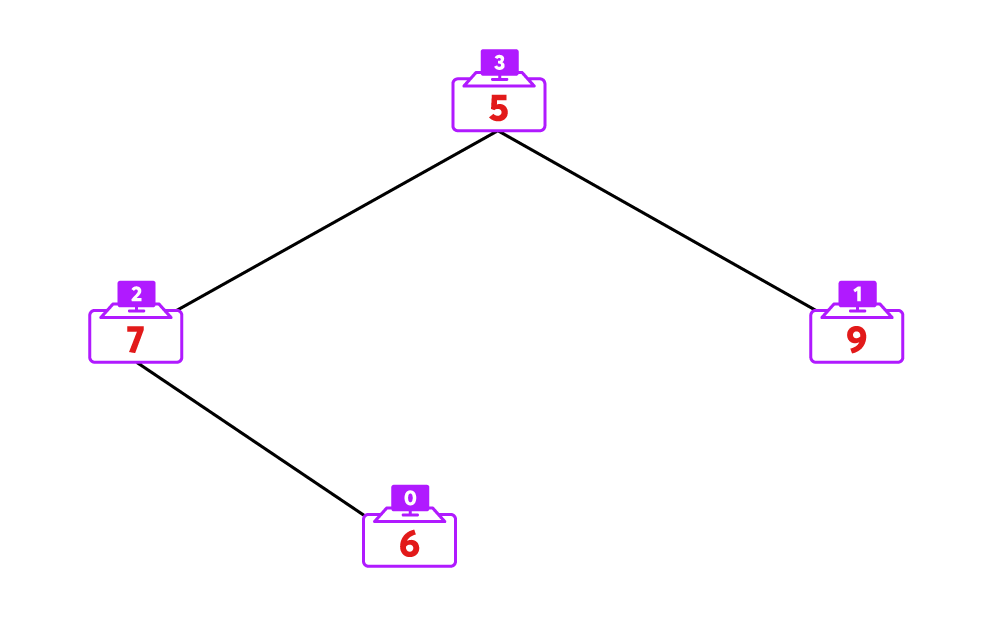
| **k** | **num** | **links** | **result** |
| --- | --- | --- | --- |
| 3 | [12, 30, 1, 8, 8, 6, 20, 7, 5, 10, 4, 1] | [[-1, -1], [-1, -1], [-1, -1], [-1, -1], [8, 5], [2, 10], [3, 0], [6, 1], [11, -1], [7, 4], [-1, -1], [-1, -1]] | 40 |
| 1 | [6, 9, 7, 5] | [[-1, -1], [-1, -1], [-1, 0], [2, 1]] | 27 |
| 2 | [6, 9, 7, 5] | [[-1, -1], [-1, -1], [-1, 0], [2, 1]] | 14 |
| 4 | [6, 9, 7, 5] | [[-1, -1], [-1, -1], [-1, 0], [2, 1]] | 9 |

입출력 예 설명

**입출력 예 #1**

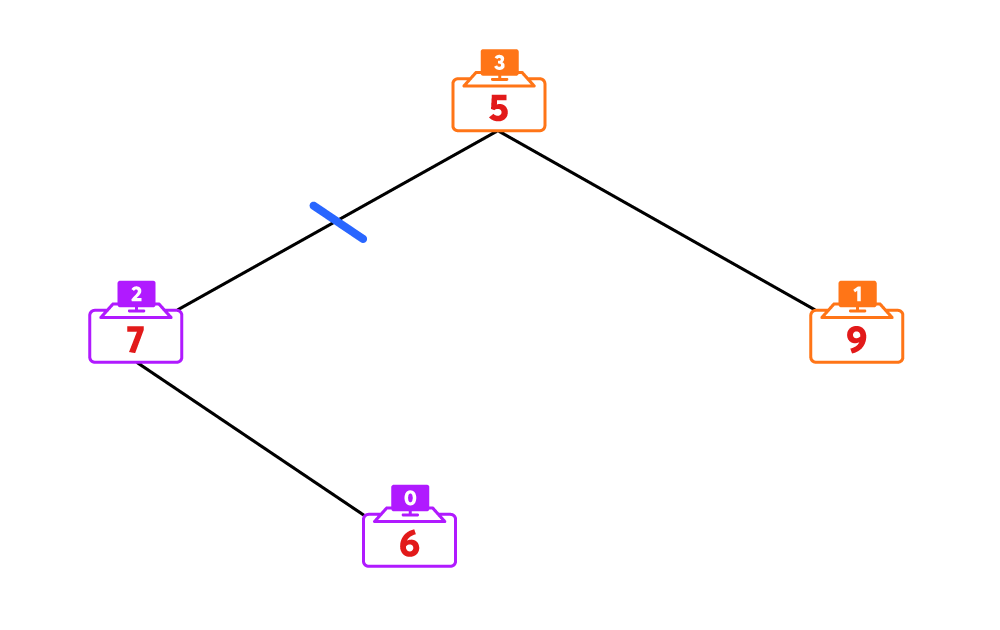
문제 예시와 같습니다.

**입출력 예 #2**



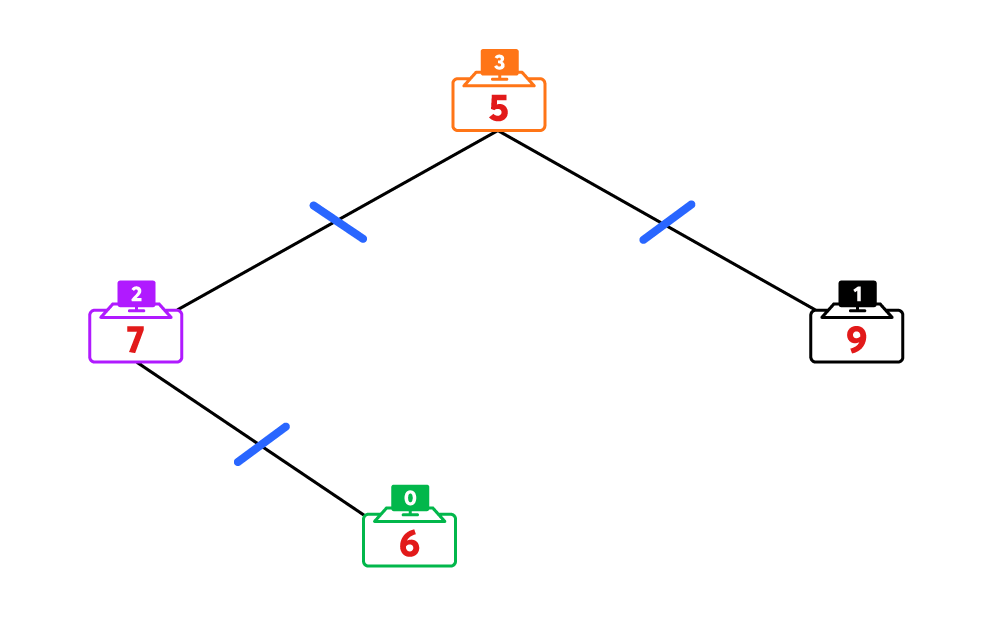
* 나눌 그룹의 수가 1개 이므로, 주어진 트리를 나눌 수 없습니다.
* 보라색 노드로 표시된 유일한 그룹의 인원은 27명입니다.

**입출력 예 #3**



* 나눌 그룹의 수가 2개 이므로, 그림과 같이 1개의 간선을 끊어서 2개의 그룹으로 나눌 수 있습니다.
* 보라색 노드로 표시된 그룹은 13명, 주황색 노드로 표시된 그룹은 14명입니다.
* 따라서, 최대 그룹의 인원은 14명입니다.

**입출력 예 #4**



* 나늘 그룹의 수가 4개 이므로, 그림과 같이 3개의 모든 간선을 끊어서 4개의 그룹으로 나눌 수 있습니다.
* 최대 그룹의 인원은 9명입니다.

제한시간 안내

* 정확성 테스트 : 10초
* 효율성 테스트 : 언어별로 작성된 정답 코드의 실행 시간의 적정 배수