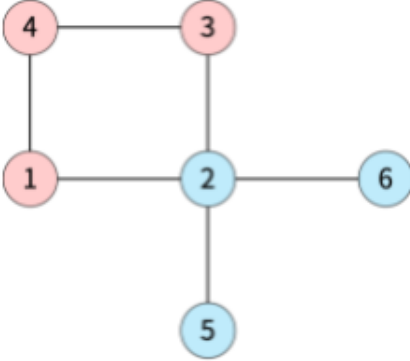
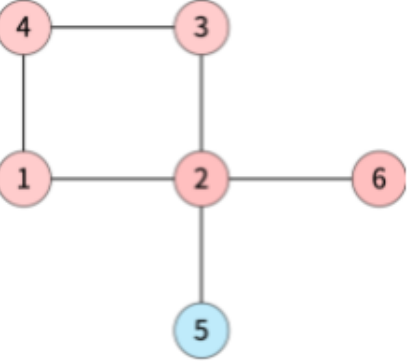
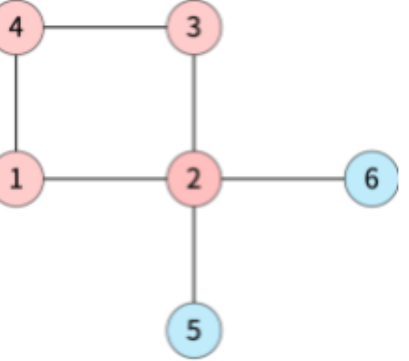
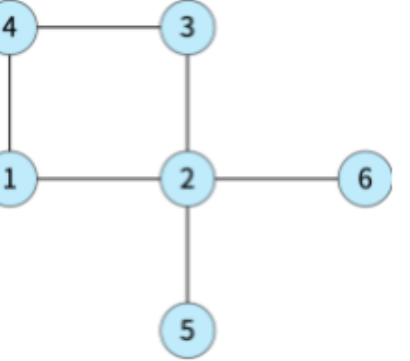


# BOJ 17471. 거리 맨더링

이진오

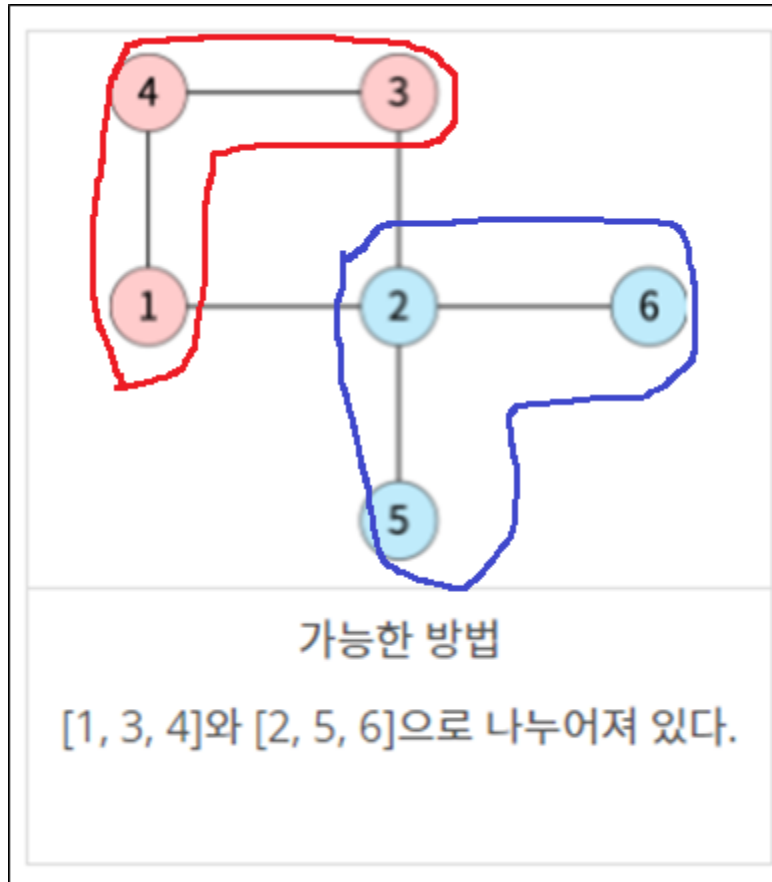
# 1. 문제 접근 / 해석

			
가능한 방법 [1, 3, 4]와 [2, 5, 6]으로 나누어져 있다.	가능한 방법 [1, 2, 3, 4, 6]과 [5]로 나누어져 있다.	불가능한 방법 [1, 2, 3, 4]와 [5, 6]으로 나누어져 있는 데, 5와 6이 연결되어 있지 않다.	불가능한 방법 각 선거구는 적어도 하나의 구역을 포함 해야 한다.

## • 부분집합 생성

- 선택된 지역 = 1정당(T) / 선택되지 않은 지역 = 2정당(F)

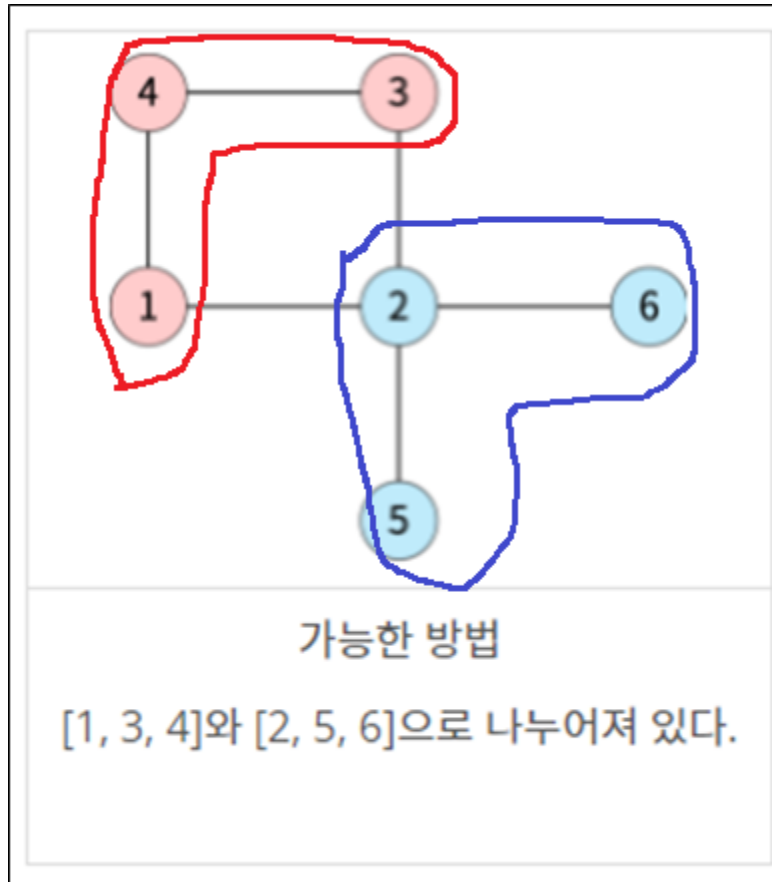
# 1. 문제 접근 / 해석



## • 연결성 확인

- 1정당 [1, 3, 4] / 2정당 [2, 5, 6]
- 각 집합이 서로 하나의 네트워크를 이루는지 확인
  - MST / BFS / DFS 등의 알고리즘을 떠올릴 수 있다.

# 1. 문제 접근 / 해석

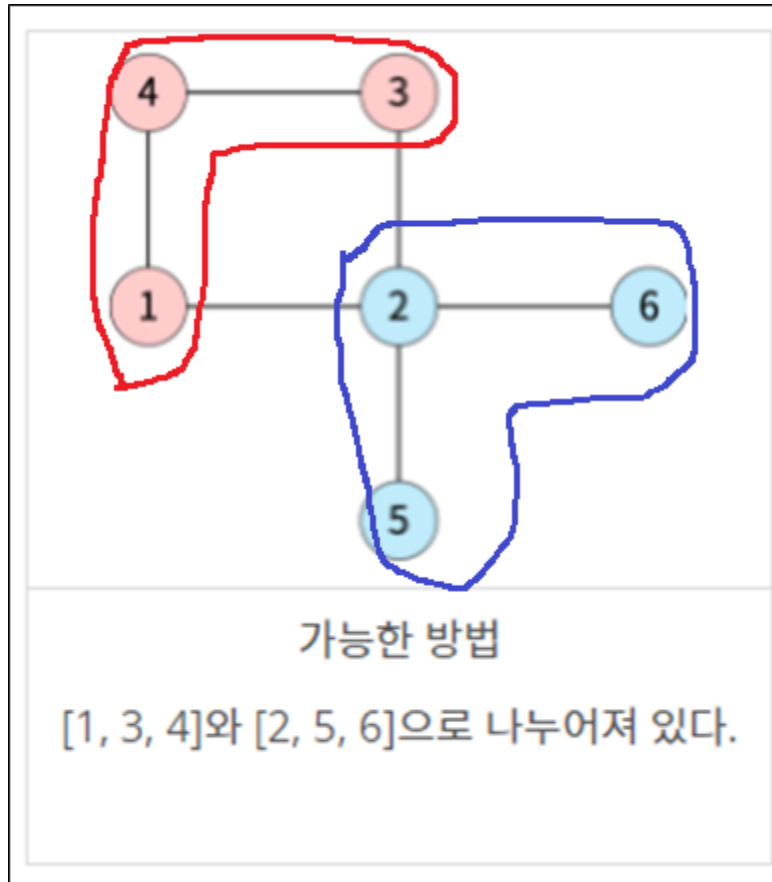


## • 연결성 확인

### • BFS

- 각 집합의 대표 지역 선정: **1정당** 대표 '1', **2정당** 대표 '2'
- 1정당에 대한 BFS
  - 초기 집합: [1]
  - 방문하지 않았으며, 1정당인 지역을 순회
- 2정당에 대한 BFS
  - 초기 집합: [2]
  - 방문하지 않았으며, 2정당인 지역을 순회

# 1. 문제 접근 / 해석

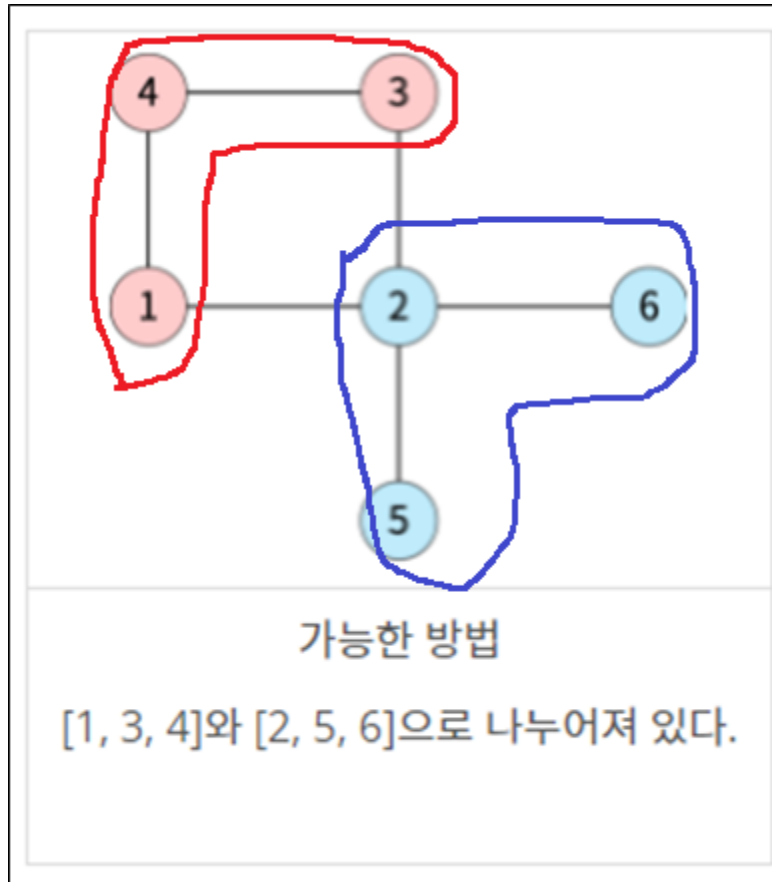


- BFS

- ※ 주의사항

- 각 BFS에 대해 방문 처리 배열(visited)을 따로 두어야 한다.
    - 1정당을 확인할 때 사용한 visited 배열을 2정당 확인 시 또 사용할 경우, 제대로 동작하지 않을 수 있다.

## 2. BFS 구현



※ 규칙: True: 1정당, False: 2정당

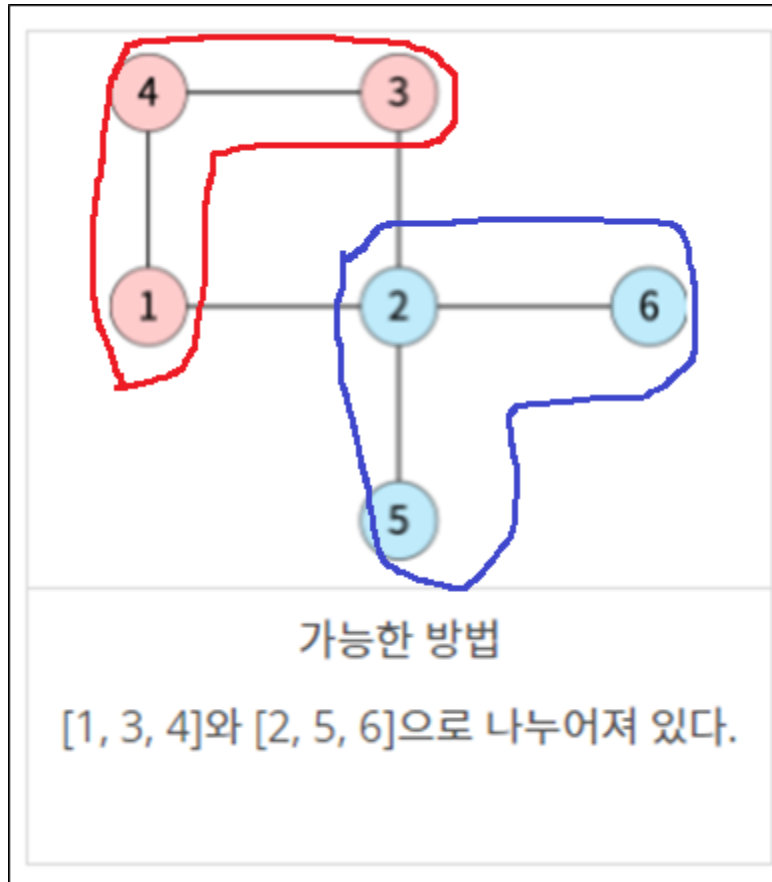
### ○ 1정당 BFS

- visited 배열의 값이 True인 경우에만 방문
- 방문 시, False로 변환!!

### ○ 2정당 BFS

- visited 배열의 값이 False인 경우에만 방문
- 방문 시, True로 변환!!

## 2. BFS 구현



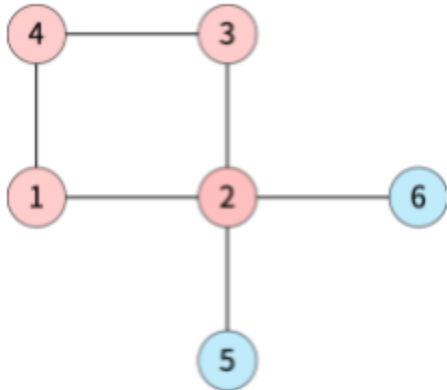
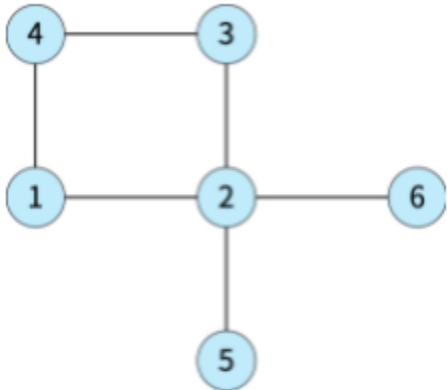
※ 규칙: True: 1정당, False: 2정당

	[1]	[2]	[3]	[4]	[5]	[6]
visited	True	False	True	True	False	False
복사						
	[1]	[2]	[3]	[4]	[5]	[6]
visited1	True	False	True	True	False	False
[1] 방문 시						
	[1]	[2]	[3]	[4]	[5]	[6]
visited1	False	False	True	True	False	False

## 2. BFS 구현

### 불가능한 경우!!

- 각 BFS 이후, 방문하지 못한 지역구가 존재
- 전체 지역구의 정당이 같을 때

	
<p>불가능한 방법</p> <p>[1, 2, 3, 4]와 [5, 6]으로 나누어져 있는데, 5와 6이 연결되어 있지 않다.</p>	<p>불가능한 방법</p> <p>각 선거구는 적어도 하나의 구역을 포함해야 한다.</p>



### 3. 문제 리뷰

- 부분 집합 코드 블록 구현
  - 지역의 개수  $N \leq 10$
  - 총 경우의 수  $\leq 1024$
- 생성된 각 부분 집합에 대하여, 문제의 요구사항 적용
  - 부분 집합이 하나의 네트워크로 연결돼 있는가?
  - 부분 집합1의 합과, 부분 집합2의 합의 차이의 최소를 구해라

## 4. 여담

Q. BFS / DFS를 먼저 수행하고, 그 이후에 헤아린다.

A. BFS / DFS는 그래프 순회 순서가 정해져 있어, 가능한 전체 집합을 헤아리기 어려움

Q. 각 네트워크의 연결 여부를 서로소 집합으로 판별?

A. 생성된 매 부분집합마다, makeSet 연산 진행

B. 1 ~ N의 노드를 순회: i번째 노드 – i번째 노드와 연결된 모든 노드에 대해 union 연산 진행

C. union 연산 진행 시, 같은 정당일 때에만 진행

## 5. 다른 구현 방법

- 부분집합 => 조합
  - N개의 지역 중 R개의 지역을 선택하는 조합 문제로 해석할 수 있다.
- BFS => DFS
  - 각 네트워크를, DFS를 통해 순회하는 로직을 구현할 수 있다.
- 인접 행렬 & 조합 & DFS 구현이 가장 빠르다. (백준 기준)