

BOJ. 1717 - 집합의 표현

0. 의의

- 서로소 집합(Disjoint set) 개념 / 구현 복습

1. 문제 해석

집합의 표현

성공 스페셜 저지

☆

4 골드 IV

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	128 MB	56454	17855	10856	28.446%


문제

초기에 $\{0\}, \{1\}, \{2\}, \dots, \{n\}$ 이 각각 $n+1$ 개의 집합을 이루고 있다. 여기에 합집합 연산과, 두 원소가 같은 집합에 포함되어 있는지를 확인하는 연산을 수행하려고 한다.

집합을 표현하는 프로그램을 작성하시오.

입력

첫째 줄에 $n(1 \leq n \leq 1,000,000)$, $m(1 \leq m \leq 100,000)$ 이 주어진다. m 은 입력으로 주어지는 연산의 개수이다. 다음 m 개의 줄에는 각각의 연산이 주어진다. 합집합은 $0 \ a \ b$ 의 형태로 입력이 주어진다. 이는 a 가 포함되어 있는 집합과, b 가 포함되어 있는 집합을 합친다는 의미이다. 두 원소가 같은 집합에 포함되어 있는지를 확인하는 연산은 $1 \ a \ b$ 의 형태로 입력이 주어진다. 이는 a 와 b 가 같은 집합에 포함되어 있는지를 확인하는 연산이다. a 와 b 는 n 이하의 자연수 또는 0 이며 같을 수도 있다.

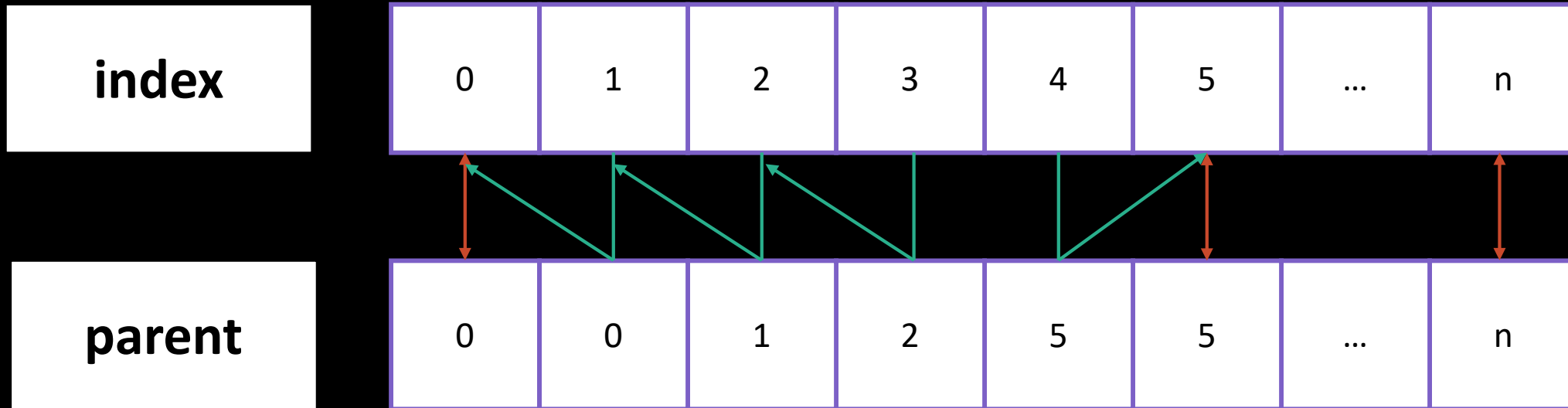
 **Disjoint set (= Union-find)**

2. Disjoint set (= Union-find)

- 접근

- 문제 내에 다수의 집합이 존재할 경우
- 다수의 집합이, 원소를 공유하지 않을 경우
- 집합을 합치는 연산(union)을 수행해야 하는 경우
- 원소가 주어졌을 때, 집합을 찾는 연산(find)을 수행해야 하는 경우

3. 자료 구조



- $\text{parent}[\text{index}] = \text{index}$ 번 원소의 부모 노드
- 배열의 값이, 곧 부모 노드를 가리키는 구조
→ 바로 이해하기 어려운 구조

4. Union, Find

```
static int[] p; // p[i] = i번 노드의 부모 노드를 가리킨다.

/**
 * @param i
 * @return i번 노드가 속한 집합의 대표자
 */
public static int find(int i) {
    // i번 노드가 곧 부모 노드인 경우를, 집합의 대표자로 규정
    if (i == p[i])
        return i;

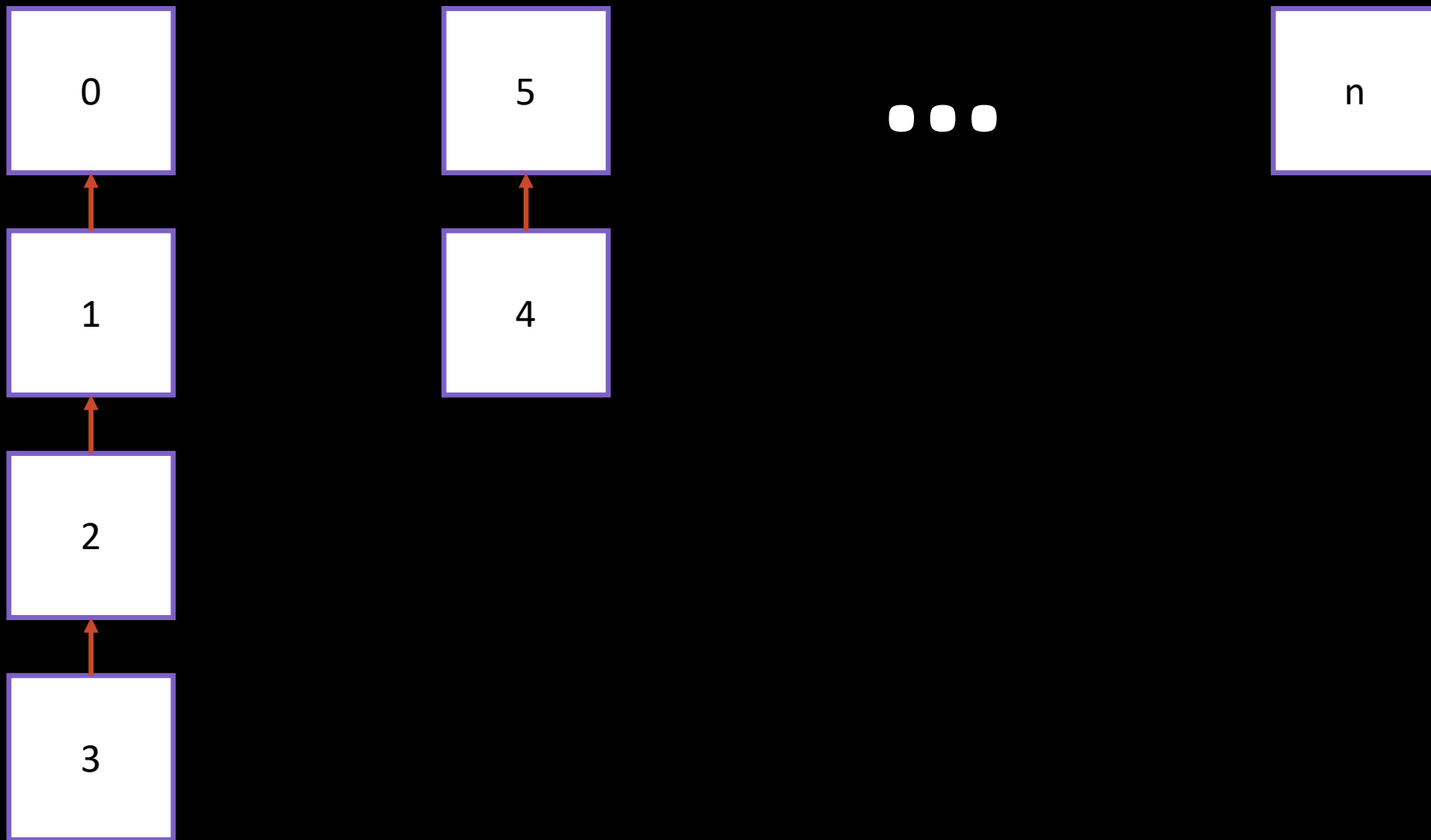
    // i번 노드의 부모 노드를, i번 노드가 속한 집합의 대표자로 변환
    return p[i] = find(p[i]);
} // end of find
```

```
/**
 * @param i
 * @param j
 * i번 노드가 속한 집합과, j번 노드가 속한 집합에 합집합 연산을 진행
 */
public static void union(int i, int j) {
    // i번 노드가 속한 집합의 대표자, j번 노드가 속한 집합의 대표자
    int rootI = find(i);
    int rootJ = find(j);

    // 두 집합이 동일하다면, 그대로 종료
    if (rootI == rootJ)
        return;

    // 두 집합이 같지 않다면, 하나의 집합을 다른 집합에 속하도록 변경
    p[rootI] = rootJ;
} // end of union
```

5. 트리(1)



5. 트리(2)

- 일반적으로 트리 자료구조를 사용하는 경우
 - 트리의 루트 노드에서, 자식 노드를 찾는 경우
 - ➔ 완전 이진 트리와 같이, 높이 $h = \log(n)$ 인 경우가 가장 효율적이다

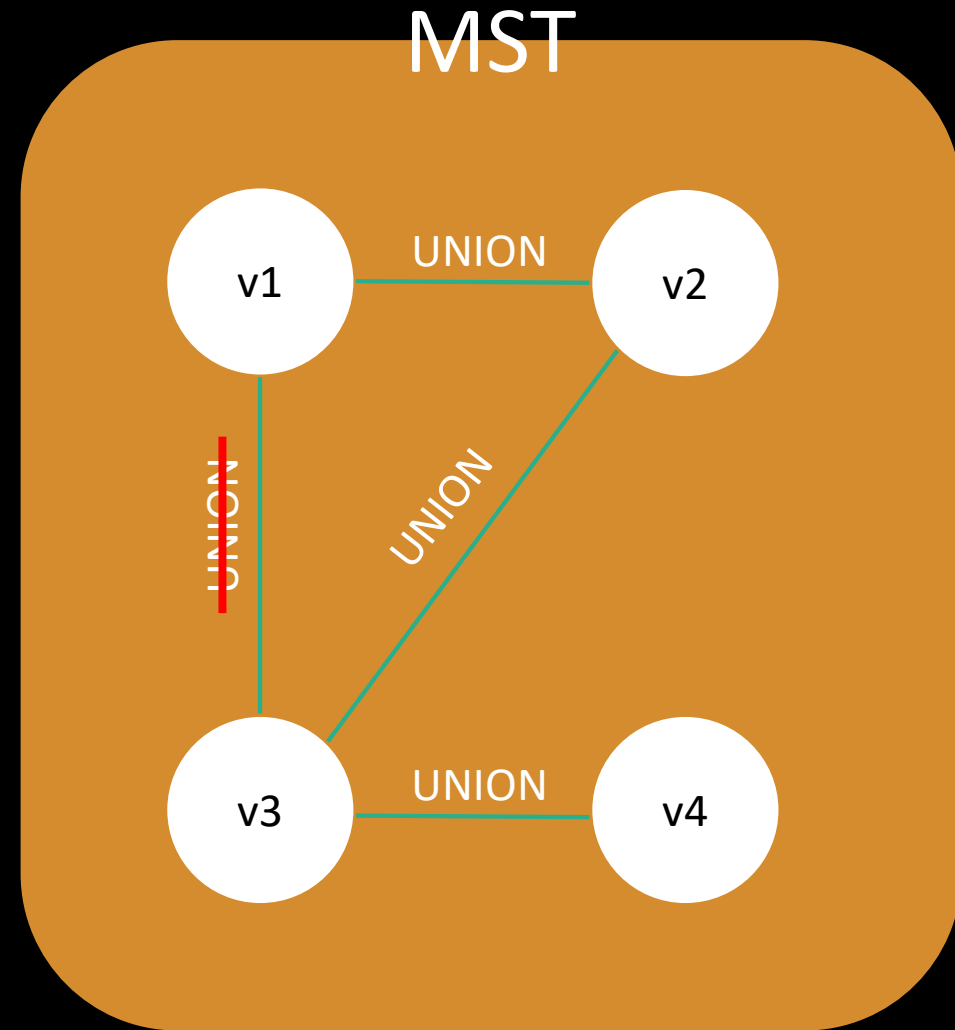
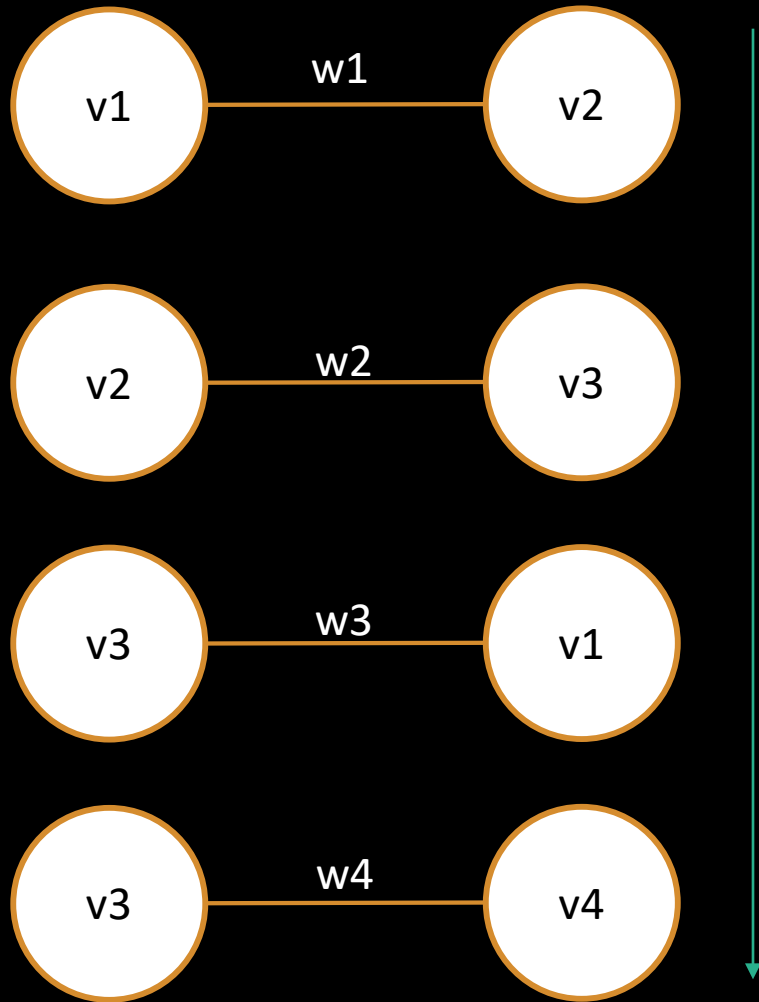
5. 트리(3)

- 서로소 집합에서 트리 자료구조를 특이하게 표현하는 이유
 - 자식 노드에서, 트리의 루트 노드를 찾는 경우 (find)
 - 트리의 높이(rank)가 낮으면 낮을 수록 좋다.
 - ➔ 완전 이진 트리보다, 깊이가 1인 트리가 더 효율적이다

6. KRUSKAL

- MST (Minimum Spanning Tree)
 - 주어진 그래프에서 전체 노드를 포함하는 최소 비용의 트리를 구성하는 문제
 - 트리
 - 최소 비용 그래프는, v 개의 정점과 $v-1$ 개의 간선으로 이루어진다.
 - v 개의 정점과 $v-1$ 개의 간선으로 이루어진 그래프는, 트리이다.
 - 네트워크를 구상하는 문제

6. KRUSKAL



7. 여담

- 서로소 집합(Disjoin set, Union-find)
 - 최소 스패닝 트리(MST) 문제를 풀 때 유용하게 활용 가능 (KRUSKAL 알고리즘!)
 - PRIM 알고리즘 활용 시 필요 없지만, PRIM은 구현이 까다롭다