10. Disjoint Set

Div. 3 알고리즘 스터디 / 임지환

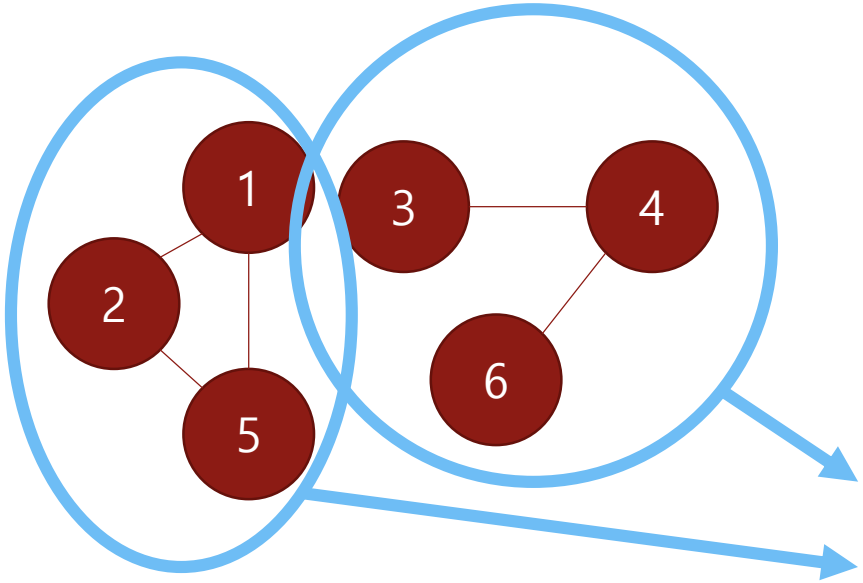
Distjoint Set

- 직역하면 "서로소 집합", 혹은 "상호배타적 집합"
- Union-Find로도 불림
- forest형태의 자료구조, 즉 기본적으로 "트리"

#11724 연결요소의 개수

- 입력 예시 :

6 5
1 2
2 5
5 1
3 4
4 6



두 집합이
상호 배타적
(Disjoint)

But 각각이 Tree는 아닌데..?

#11724 연결요소의 개수

- 입력 예시 :

  6 5

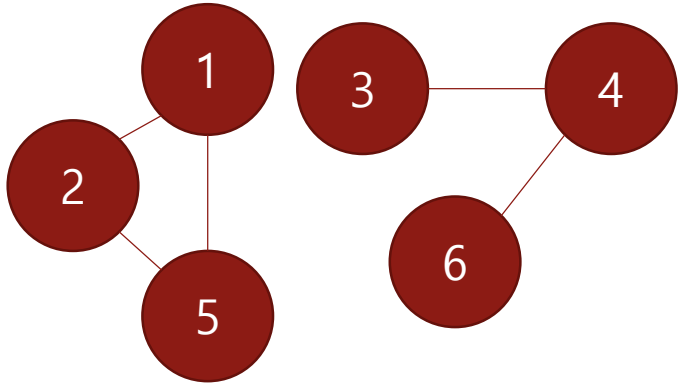
  1 2

  2 5

  5 1

  3 4

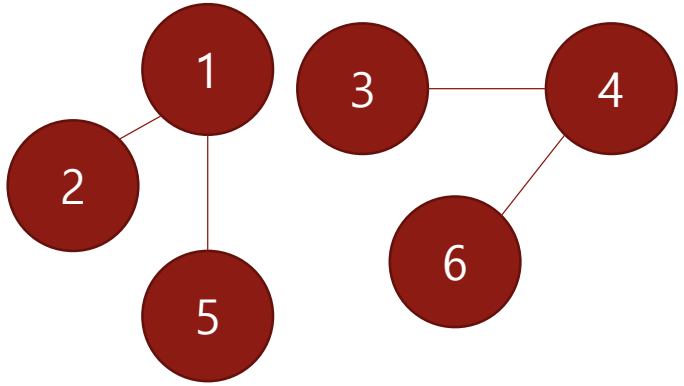
  4 6



#11724 연결요소의 개수

- 입력 예시 :

```
6 5
1 2
2 5
5 1
3 4
4 6
```



(2,5) or (1,2) or (1,5)이 끊어져도 {1, 2, 5}는 한 연결요소

Union-Find

- 연산이 단 2개!

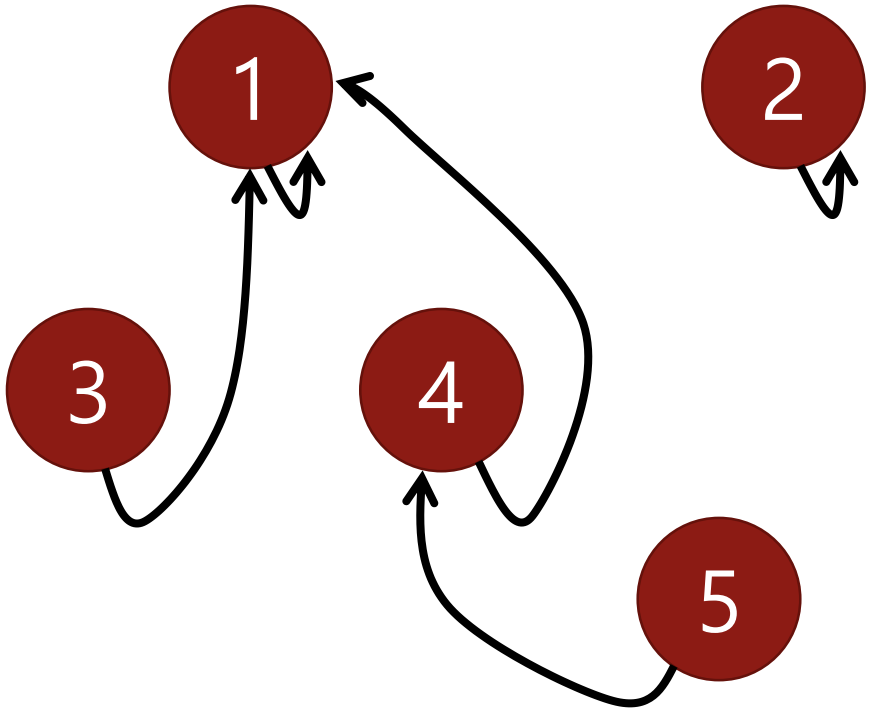
  Union operation

  Find operation

- 정말 저게 끝이다..

Find Operation

- Union-Find에서의 Find연산은 해당 노드의 "~~부모~~"를 return
  조상, 혹은 대푯값

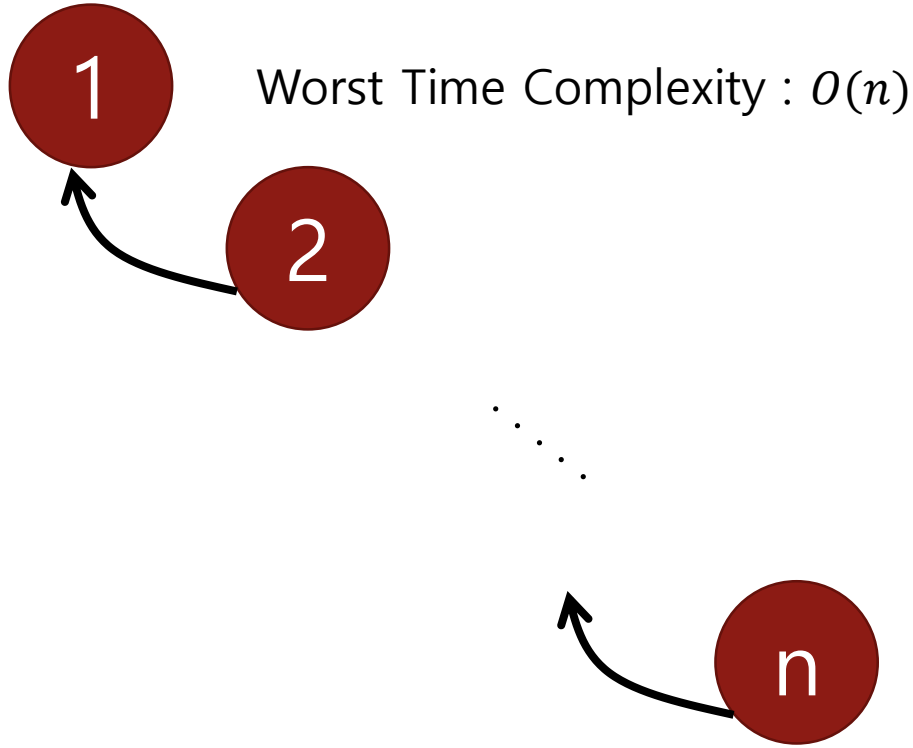


Find(1) = 1
Find(2) = 2
Find(3) = 1
Find(4) = 1
Find(5) = 1

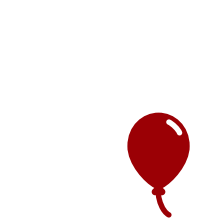
Find Operation

parent_of[i] := ‘i’의 부모

```
int find (int index) {
    if (parent_of[index] == index)
        return index;
    else return ( find(parent_of[index]) );
}
```



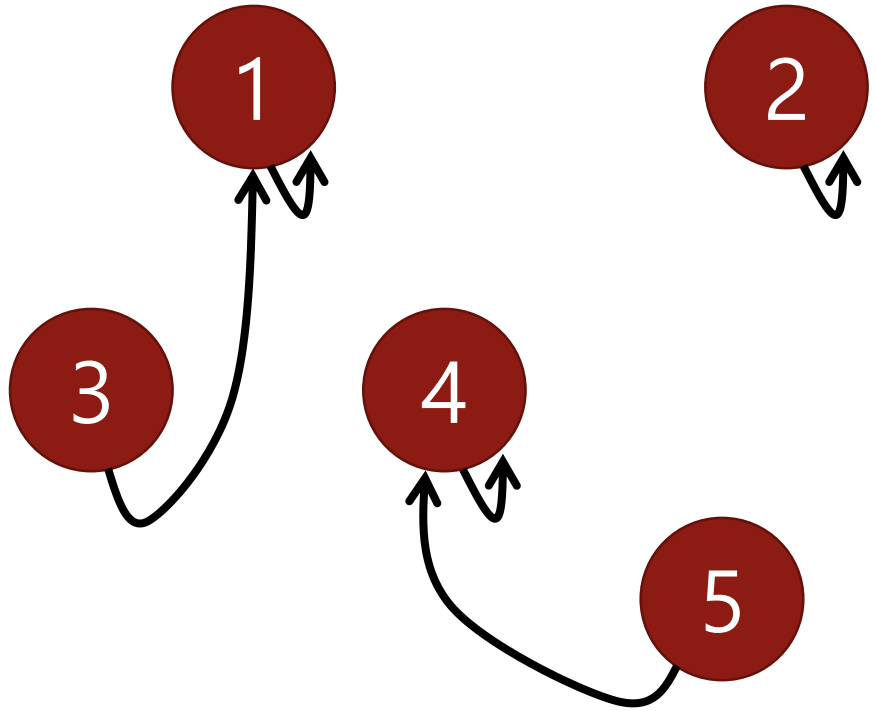
Worst Time Complexity : $O(n)$



Union Operation

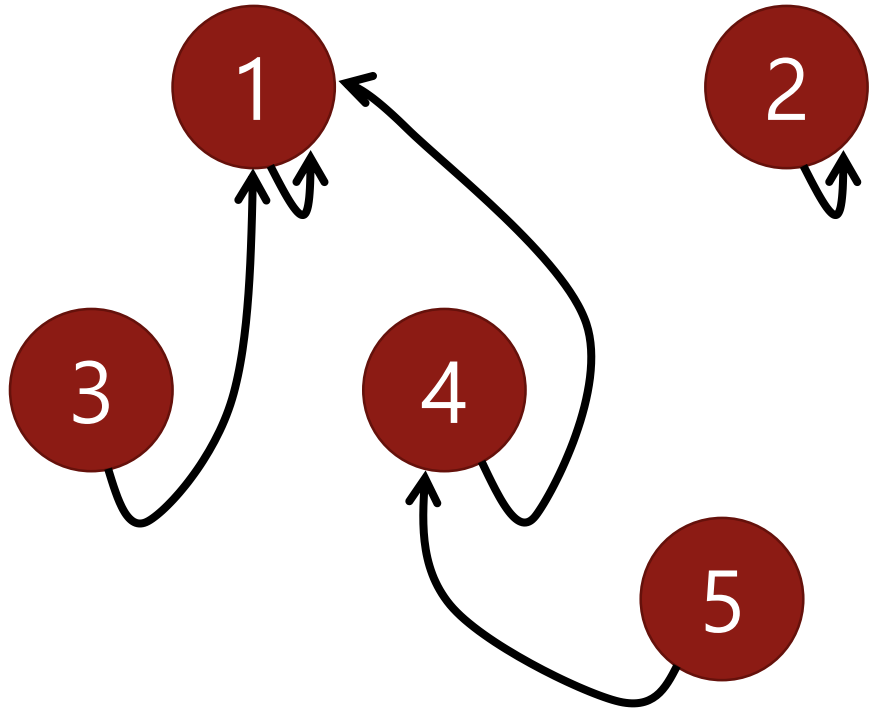
- 상호배타적인 두 원소를 한 집합으로 만드는 연산

Union(1, 4)



Union Operation

- 상호배타적인 두 원소를 한 집합으로 만드는 연산

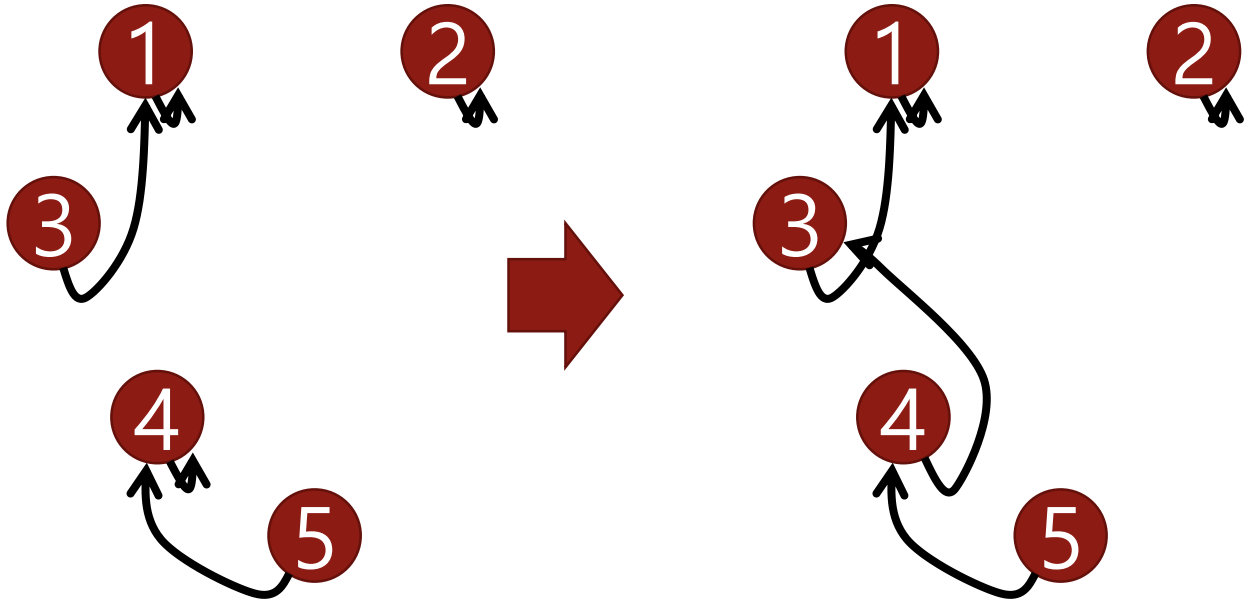


Union Operation

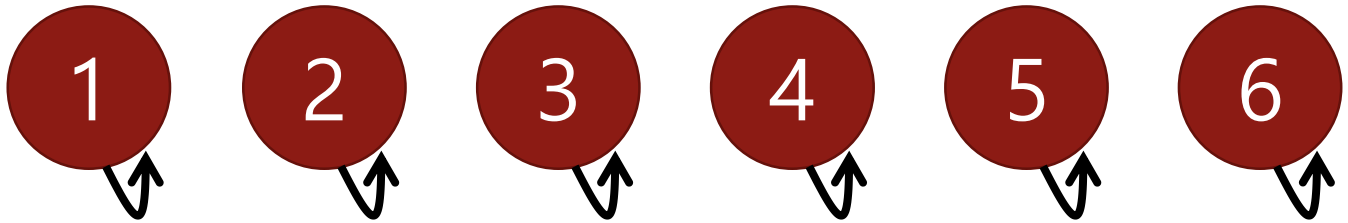
parent_of[i] := ‘i’의 부모

```
void merge (int a, int b) {
    a = find(a), b = find(b);
    if (a == b) return;
    parent_of[b] = a;
}
```

Union 연산 이후에 find(5)를 호출한다면?

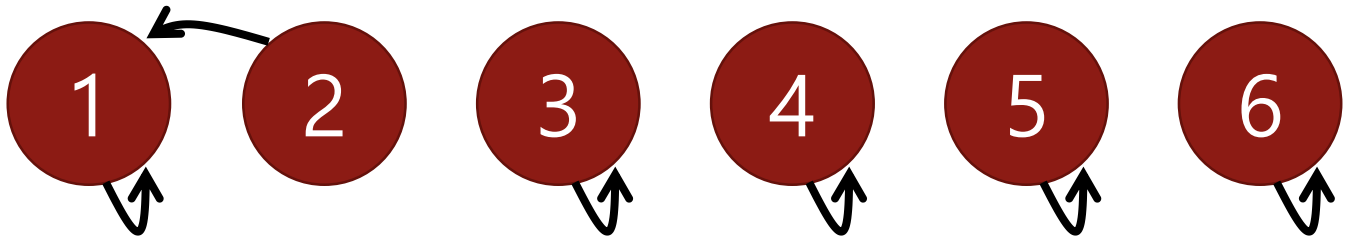


Example



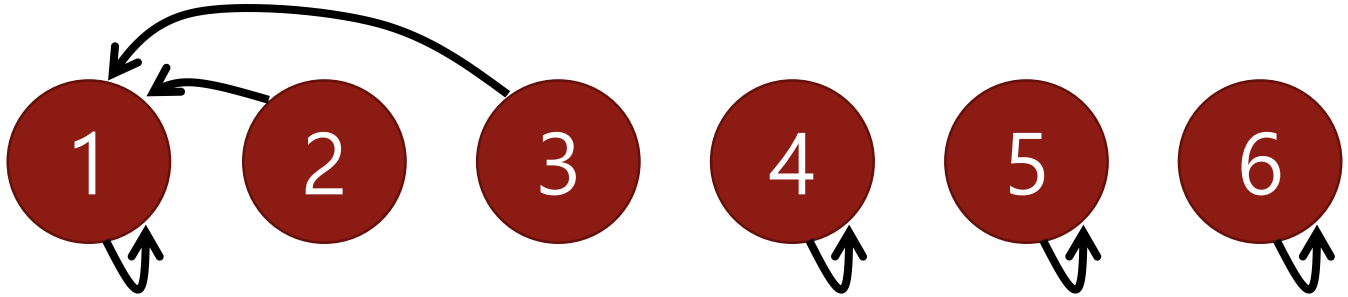
초기 상태 : 모든 노드의 부모가 자기 자신

Example



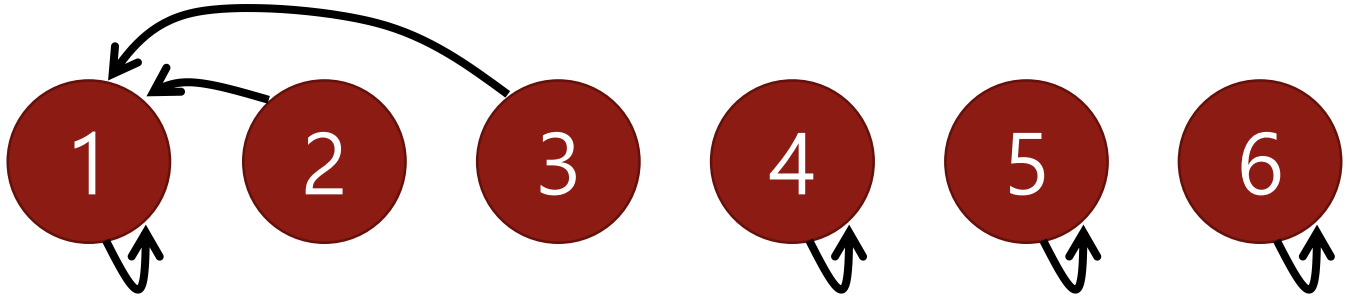
union(1,2)

Example



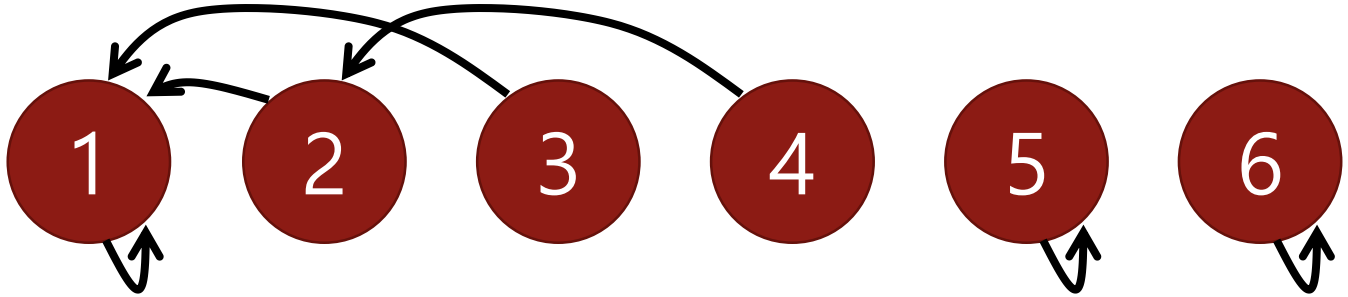
union(1,2)
union(1,3)

Example



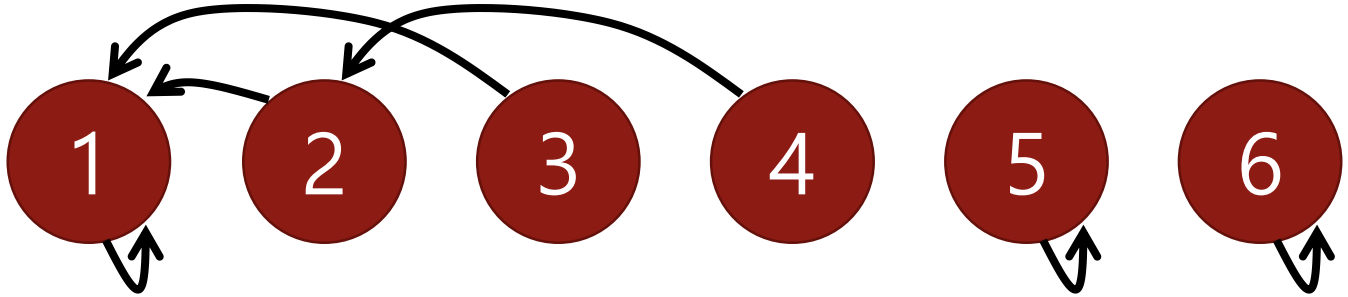
union(1,2)
union(1,3)
find(3) => "1"

Example



union(1,2)
union(1,3)
find(3) => "1"
union(2,4)

Example



union(1,2)
union(1,3)
find(3) => "1"
union(2,4)
find(4) => "1"

#11724 연결요소의 개수

- 입력 예시 :

  6 5

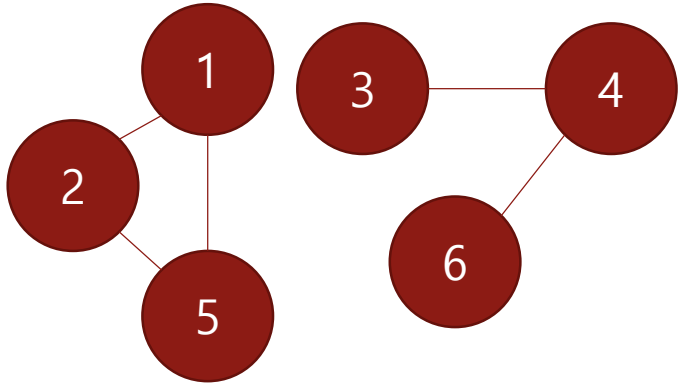
  1 2

  2 5

  5 1

  3 4

  4 6



#11724 연결요소의 개수

- 간선 정보로 주어지는 (u, v) = u와 v가 한 집합에 속해 있음
- 간선 정보가 모두 주어진 후 연결요소의 개수를 세는 방법
  1) 최초 상태의 연결요소의 개수 : 노드 개수
  2) union이 일어날 때마다 연결요소의 개수가 하나씩 줄어듬

     ⇒union이 일어나는 횟수

#11724 연결요소의 개수

```
bool merge(int a, int b) {
    if ((a = find(a)) == (b = find(b))) return false;
    par[b] = a;
    return true;
}
```

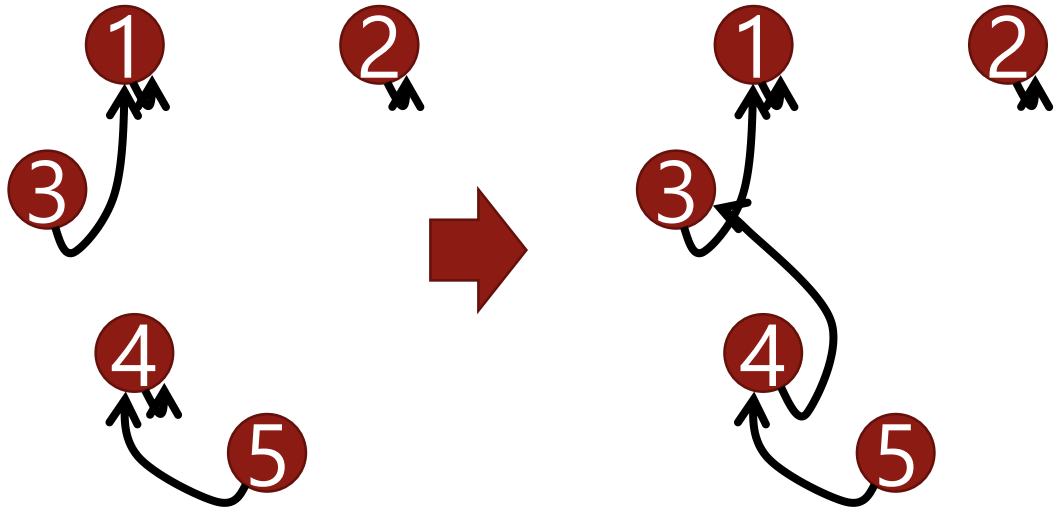
merge함수가 꼭 void type일 필요는 없다.
성공적으로 merge를 수행할 경우 true를,
아닌 경우(이미 한 group인 경우) false를 return한다.

http://boj.kr/c1ea4bdd29894bb099c6e024de551603

Union-Find 최적화

1. Union by Rank

   Union 연산에서 노드 삽입 시, 두 집합을 구성하는 트리의 높이를 판별하여, 높이가 더 높은 트리의 루트 노드에 높이가 낮은 트리를 병합



Union-Find 최적화

1. Union by Rank

   높이가 1인 집합의 경우 : 1개의 원소

   높이가 2인 집합의 경우 : 최소 2개의 원소

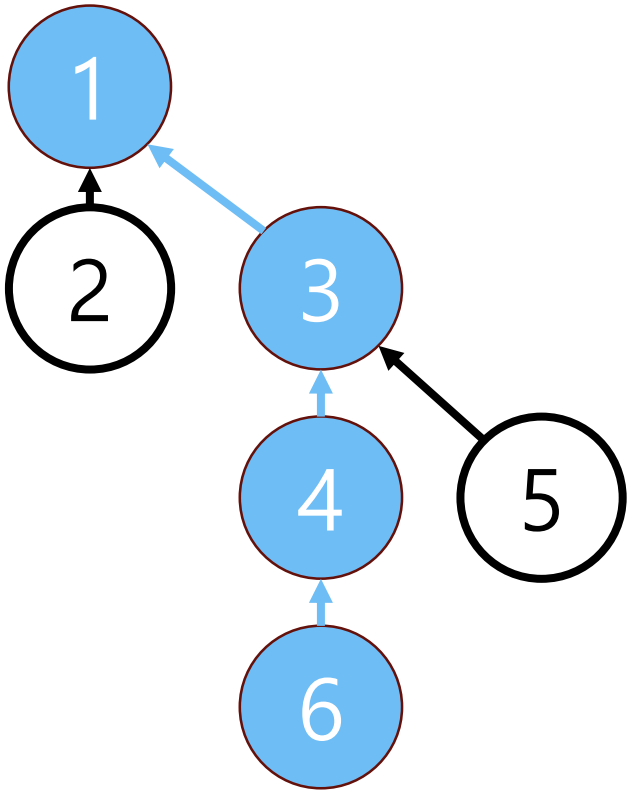
   높이가 n+1인 집합이 만들어지려면?

   ⇒같은 높이 n인 두 트리가 합쳐져야 함

Time complexity : $O(logn)$

Union-Find 최적화

2. Path Compression



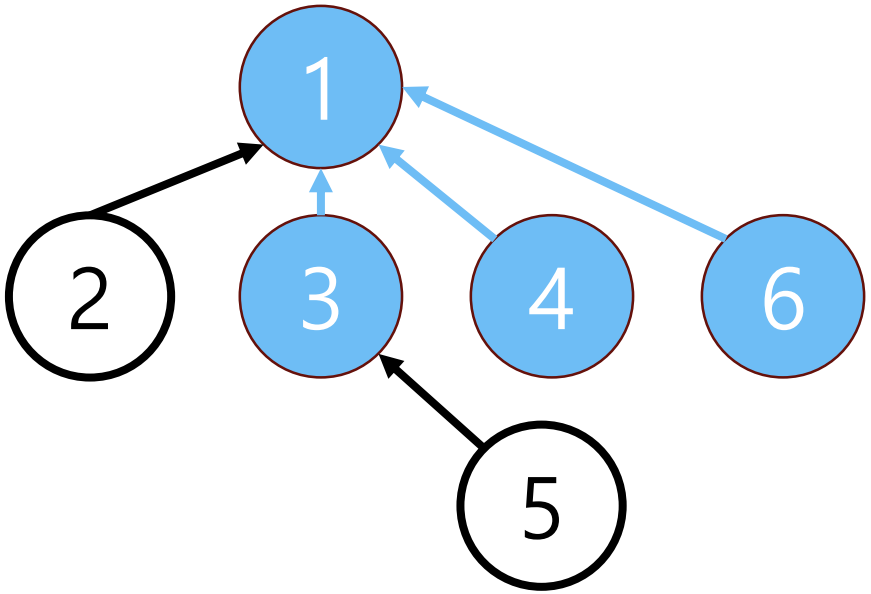
find(6)이 호출되었을 경우

3, 4, 6이 집합 1에 속한다.

이거를 다시 검사할 필요가 있을까?

Union-Find 최적화

2. Path Compression



한번의 find연산 이후 Tree의 구조를 변경하여, 트리의 전체 높이를 감소시킨다.

다음 연산에 의해 이후 발생하는 find연산에 소요되는 시간이 감소

Time complexity : $O(logn)$

Total Time Complexity

Union By Rank와 Path Compression을 수행하였을 경우,

Union연산과 Find연산의 시간 복잡도는 $O(\alpha(n))$ 으로 알려져 있다.

여기서 $\alpha(n)$ 은 Inverse Ackermann Function으로

매우 느리게 증가하여.....

따라서 거의 $O(1)$에 근접하는 시간 복잡도를 가진다.

#13244 트리

간선의 정보들이 주어질 때 만들어진 그래프가 트리인지 트리가 아닌지 판별하는 문제

트리의 조건

- 간선 개수 = 정점 개수 – 1
- 한 연결요소에 속하면서 사이클이 없는 경우

#13244 트리

- 사이클 판정을 어떻게 하지?

  ⇒DFS를 쓴다면, 방문 여부 체크를 통해 확인 가능

  ⇒Union-Find를 쓴다면?

조상이 같은 두 노드에 대해 Union연산을 수행한다면
사이클 발생!

http://boj.kr/2ee193bc26dc4bf393db42832070ef55

#16562 친구비 2018 Sogang Programming contest(Master)

- 친구가 되려면 친구비를 내면 된다.
- "친구의 친구는 친구"이기 때문에, 친구비를 적게 내는 사람을 적절히 골라 친구비를 내면 적은 비용으로 친구를 만들 수 있다.

#16562 친구비 2018 Sogang Programming contest(Master)

- 만들어진 disjoint set마다 한 명만 선택하면 된다는 것은 알겠는데, 그중 비용이 가장 적은 사람은 어떻게 선택해야 하지?
- Union 단계에서 비용이 가장 적은 사람을 대푯값으로 설정

#16562 친구비 2018 Sogang Programming contest(Master)

```
bool merge(int a, int b) {
    a = find(a); b = find(b);
    if (a == b) return false;
    if (price[a] > price[b]) par[a] = b;
    else par[b] = a;
    return true;
}
```

b의 비용이 더 적으므로,
b를 대푯값으로 설정

#10775 공항

- 게이트의 정보 g_i가 주어졌을 때, 1~g_i 중 어느 곳에든 도킹 가능하다.
- g_1 = 2, g_2 = 2로 주어진다면, 1번째 비행기를 1번에, 2번째 비행기를 2번째에 도킹 가능하다.(바뀌어도 상관이 없다.)
- g_i값이 k로 주어진다면, i≠j 인 j에 대하여 g_j=k인 j는 최대 k개가 나올 수 있다.

http://boj.kr/1d5177e78cc34078aa71470b68a2da3f

Problem Set

- 11724 연결 요소의 개수
- 13244 Tree
- 1717 집합의 표현
- 1976 여행 가자
- 4195 친구 네트워크
- 16562 친구비
- 10775 공항(*)
- 16402 제국(**)
- 3830 교수님은 기다리지 않는다(**)
- 13306 트리(**)