Union Find

발표자 김윤재

What is union find?

- 유니온 파인드는 그래프 알고리즘으로 , 여러 노드가 존재할 때, 두 노드를 선택해서 현재 두 노드가 서로 같은 그래프에 속하는지 판별하는 알고리즘이다.
- Union/Find 2가지 연산으로 이루어져 있는데,
  Find는 X가 어떤 집합에 포함되어 있는지를 찾는 연산이며,
  Union은 X가 포함되어 있는 집합을 다른 집합이랑 합치는 연산이다.

- 유니온 파인드가 시작할 때는, 자기 자신의 부모는, 자기 자신으로 설정합니다.

Union find use tree

- 유니온 파인드 알고리즘은 트리 자료구조를 사용합니다.
- 만약 두 개의 트리를 Union(병합) 할때, 트리의 깊이랑 넓이가 더 넓은 tail에 더 작은 노드를 붙히는 것이 일반적입니다,
  그래서 rank 배열을 설정하여, 더 큰 tree에 더 작은 tree를 병합할 수 있도록 해주는 게 효율적입니다.

Find

```
public static
int find(int x)
{
    if(x == parent[x])
    return x;
    else
    return parent[x] =
    find(parent[x]);
    }
```

파인드 함수 같은 경우,
만약 인자가 자기 자신이라면? (자기 자신이 부모 노드인 경우)
자기 자신의 값을 반환합니다.

만약 인자가 자기 자신이 아니라면?(자신의 부모 노드가 있는 경우)
Find 함수를 한번 더 실행합니다.
그러면 결국 부모의 노드를 계속 탐색하며, 루트 노드의 값이 반환됩니다.

Union

```
Public static void union(int x,, int y){
        x = find(x);
        y=find(y);
        if(x==y)
                return;

        if(rank[x]<rank[y]){
                parent[x]=y;
                rank[y]+=rank[x];
        }else{
                parent[y]=x;
                rank[x]+=rank[y];
        }
```

Rank 배열에는 처음부터 모든 인덱스마다 1을 대입하고, rank값이 큰 그래프에, rank값이 작은 그래프를 흡수하고, rank값을 키워서, 더 가중치가 큰 그래프가 부모 그래프가 될 수 있도록 하는 코드입니다.
Rank 값이 같을 경우에는, x가 부모가 될 수 있게 설계했습니다.

문제

초기에 {0}, {1}, {2}, ... {n} 이 각각 n+1개의 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 ≤ n ≤ 1,000,000), m(1 ≤ m ≤ 100,000)이 주어진다. m은 입력으로 주어지는 연산의 개수이다. 다음 m개의 줄에는 각각의 연산이 주어진다. 합집합은 0 a b의 형태로 입력이 주어진다. 이는 a가 포함되어 있는 집합과, b가 포함되어 있는 집합을 합친다는 의미이다. 두 원소가 같은 집합에 포함되어 있는지를 확인하는 연산은 1 a b의 형태로 입력이 주어진다. 이는 a와 b가 같은 집합에 포함되어 있는지를 확인하는 연산이다. a와 b는 n 이하의 자연수 또는 0이며 같을 수도 있다.

출력

Baekjoon 1717 _ gold 4

1로 시작하는 입력에 대해서 한 줄에 하나씩 YES/NO로 결과를 출력한다. (yes/no 를 출력해도 된다)

예제 입력 1 복사

```
7 8
0 1 3
1 1 7
0 7 6
1 7 1
0 3 7
0 4 2
```

예제 출력 1 복사

```
NO
NO
YES
```

문제 해결 전략.

- 입력이 0일 때, 두 개의 집합을 union하고,
- 입력이 1일 때, 두 개의 집합의 부모를 find해서 같으면 yes, 틀리면no를 출력하는 간단한 문제. 유니온 파인드의 기초격 문제이다.
- 간단하게 union 함수와 find함수를 정석으로 구현해주면 끝.

```java
import java.util.*;
public class Unionfind {

    static int[] parent;
    static int[] rank;
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int m=in.nextInt();
        parent = new int[n+1];
        rank = new int[n+1];
        for(int i=1;i<n+1;i++){
            parent[i] = i;
            rank[i]=1;
        }

        for(int i=0;i<m;i++){
            int dis = in.nextInt();
            int a = in.nextInt();
            int b = in.nextInt();

            if(dis==0)
                union(a,b);
            else{
                if(find(a)==find(b))
                    System.out.println("YES");
                else
                    System.out.println("no");
            }
        }
    }
```

처음에 paren배열과, rank 배열을 각각 i와 1로 초기화를 시켜준 후, a=0일때 union을 진행, a=1일때 find로 부모가 같은지를 찾아준다.

감 사 합 니 다.