최소 신장 트리 MST

① Prim $O(m \log n)$ 정점 n, edge m

아무 노드에서부터 인접한 노드 중 가장 작은 간선으로 연결된 것을 추가
(사이클이 없도록)

가장 작은 edge를 찾을 때 우선순위 큐 (heap 활용) 로 구현

T_{MST} 는 MST edge set 이다.
T는 알고리즘이 만들어 가고 있는 edge set 이다.

Base. node 1개 edge 1개면 자명하게 성립.

Step. T에 k개의 edge가 들어있고, $T \subseteq T_{MST}$ 가 성립한다.
그 다음에 T에 추가되는 edge가 e 라고 하자.

Case 1. $T \cup \{e\} \subseteq T_{MST}$ 이면, 성립한다.
Case 2. $T \cup \{e\} \not\subseteq T_{MST}$ 이면 T_{MST}에 e를 추가했을 때, 사이클이 생성된다.
MST 자체가 모든 정점이 최소비용으로 연결된 트리인데,
여기없는 edge를 추가하면 당연히 사이클이 생김.
이 때, 이 사이클 안에 1. e와 다르고,
2. T와는 사이클을 이루지 않으며,
3. T와 인접한 edge
e'가 존재한다. 즉, e와 경쟁 상대 였던 edge 다.

e'의 weight에 따라서...

2-1 $w(e) = w(e')$; 또다른 정답이다.

2-2 $w(e) > w(e')$: 알고리즘이 e'를 선택했을 것이다.

2-3 $w(e) < w(e')$: T_{MST}가 정답이 아니게 된다.

$(T_{MST} - \{e'\}) \cup \{e\}$라는 더 좋은 답이 존재하게 되므로 모순!

② Kruskal $O(m \log m)$ ← 간선 정렬해주는데 듦. Union-Find연산의 상수시간

"전체" 그래프에서 가장 적은 간선을 계속추가 (사이클이 생기면 버림)

↓

사이클 판별에 Union, Find 사용.

T_{MST}가 MST의 edge set과 비교하자.

T는 알고리즘이 만들어가고 있는 edge set 이다.

Base: node 1개, edge 1개면 자명하게 성립.

Step: Prim과 같다.

⋮

이 때, 사이클을 판단하기 위해 Union-Find 를 활용.

덩어리마다 대장노드가 있고, Find 하면 그 노드가 속한 덩어리의 대장노드가 나옴.

Find 할 때, 모든 노드가 대장노드에 직접연결하게 바꾸는 Path compression으로 성능 향상 가능!