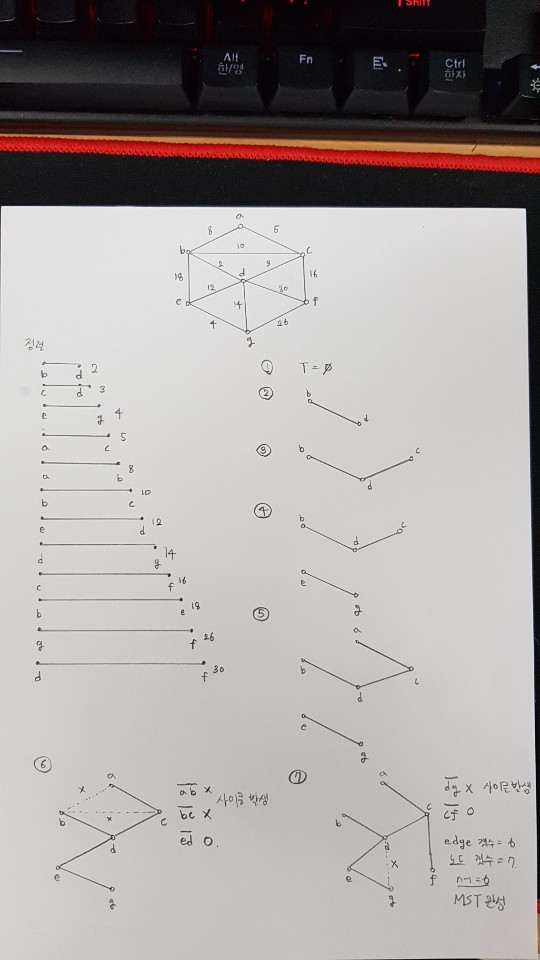
알고리즘 응용 #6과제

20144649 고한설

#1

#2

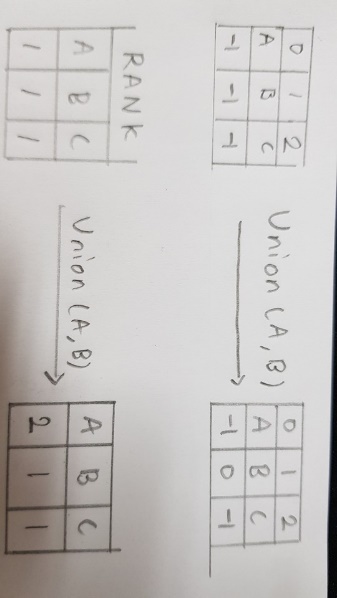
Union 과 Find 연산을 위해 트리와 같은 자료구조 형식을 사용합니다.

저는 배열의 자료구조로 트리를 표현하는 방식을 알아보겠습니다. 우선 2가지 배열을 가집니다.

하나의 배열은 부모를 가리키는 것을 표현하기위한 배열 TREE[]

하나는 트리의 개수를 위한 배열입니다. RANK[]

먼저 부모를 가리키는 배열들은 초기값을 -1을 가집니다. 처음엔 자기자신 뿐이기 때문입니다.

그러다 Union 연산을 통해 다른 집합과 합쳐진다면 RANK에 따라서 더 많은 수의 자식을 가진 트리가 작은 트리를 먹는 형태처럼 작은 트리의 ROOT 가 부모 인덱스를 가리키게 됩니다.

만일 Union 연산을 실행할 시 이와 같이 배열의 값이 해당 부모의 인덱스 값을 나타냅니다.

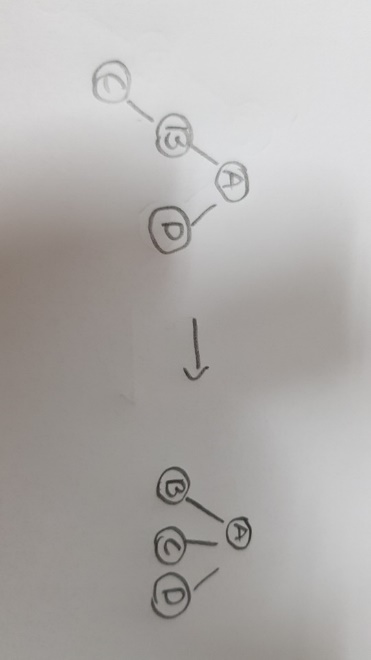
Find 연산은 해당 원소가 속한 집합의 ROOT 노드를 반환합니다. 여기선 결국 ROOT의 인덱스 죠.

Find(B) 연산을 실행할 시 부모 노드를 찾아 가면서 그 노드의 값이 (해당 인덱스의 값) -1이된다면 그 집합의 ROOT 노드인 것을 확인할 수 있습니다. 이 상태에선 0 이 return 됩니다.

찾아가는 과정에서 Find연산은 해당 트리의 부모 노드를 더 빨리 찾을 수 있도록 최적화를 합니다.

해당 부모를 찾아가면서 만난 노드들을 ROOT로 만들어주는 것이죠 트리의 높이는 줄이는 것은 탐색하는 속도도 빠르게 해줍니다.

최악의 경우 트리에서 찾는 연산 횟수는 트리의 높이 이기 때문입니다. (이진 트리일 경우)

만일 트리의 형태가 사진과 같을 때에 Find 연산을 호출한다면 Ex) Find(C)

저렇게 노드C가 B의 부모 A(ROOT)를 바로 가리키게 됩니다.

#3