비선형 자료 구조

일렬로 나열하지 않고 자료 순서나 관계가 복잡한 구조

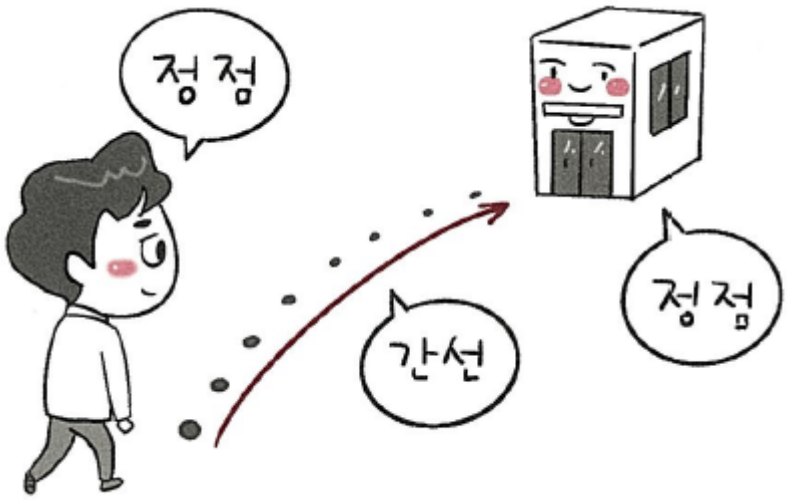
일반적으로 트리나 그래프를 말함

그래프

정점과 간선으로 이루어진 자료 구조

정점과 간선

어떠한 곳에서 어떠한 곳으로 무언가를 통해 간다고 했을 때, ‘어떠한 곳’ 은 정점 (vertex) 이 되고 ‘무언가’ 는 간선 (edge) 이 됨



indegree : 다른 정점에서 들어오는 간선의 개수

outdegree : 다른 정점으로 나가는 간선의 개수

가중치

간선과 정점 사이에 드는 비용

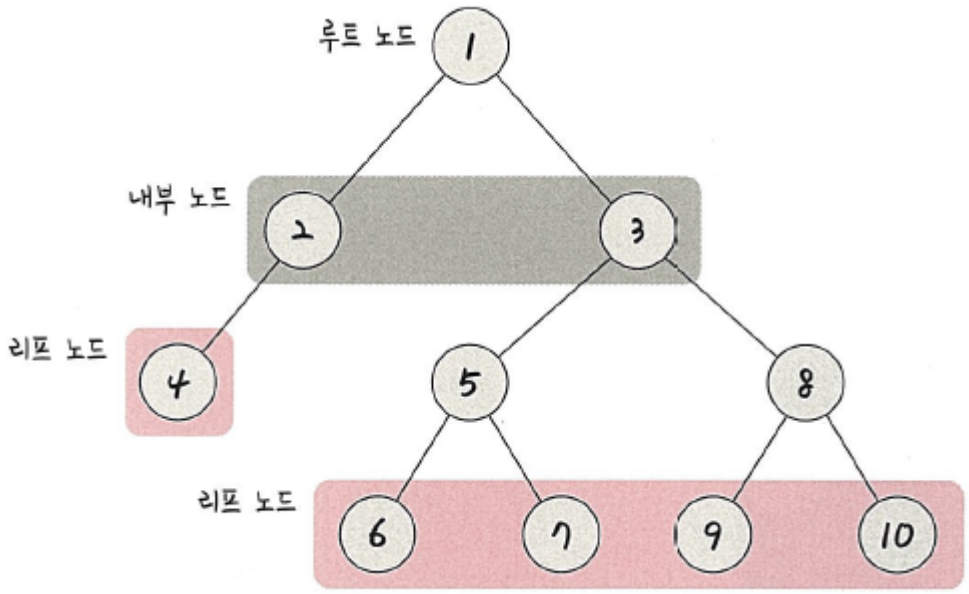
트리

그래프의 일종으로, 그래프의 특징처럼 정점과 간선으로 이루어져 있고, 트리 구조로 배열된 일종의 계층적 데이터의 집합

루트 노드, 내부 노드, 리프 노드 등으로 구성됨

트리로 이루어진 집합을 숲이라고함

특징



아래 특징들에 의해 그래프와의 차이점을 가짐

1. 부모, 자식 계층 구조를 가짐
2. V - 1 = E 라는 특징이 있음. 즉, 간선 수는 노드 수 - 1
3. 임의의 두 노드 사이의 경로는 유일무이 하게 존재함. 즉, 트리 내의 어떤 노드와 어떤 노드까지의 경로는 반드시 존재함

구성

루트 노드, 내부 노드, 리프 노드로 구성

루트 노드

트리의 가장 위에 있는 노드

탐색의 시작점이며, 일반적으로 트리 탐색 시 루트 노드를 중심으로 탐색하면 문제가 쉽게 풀리는 경우가 많음

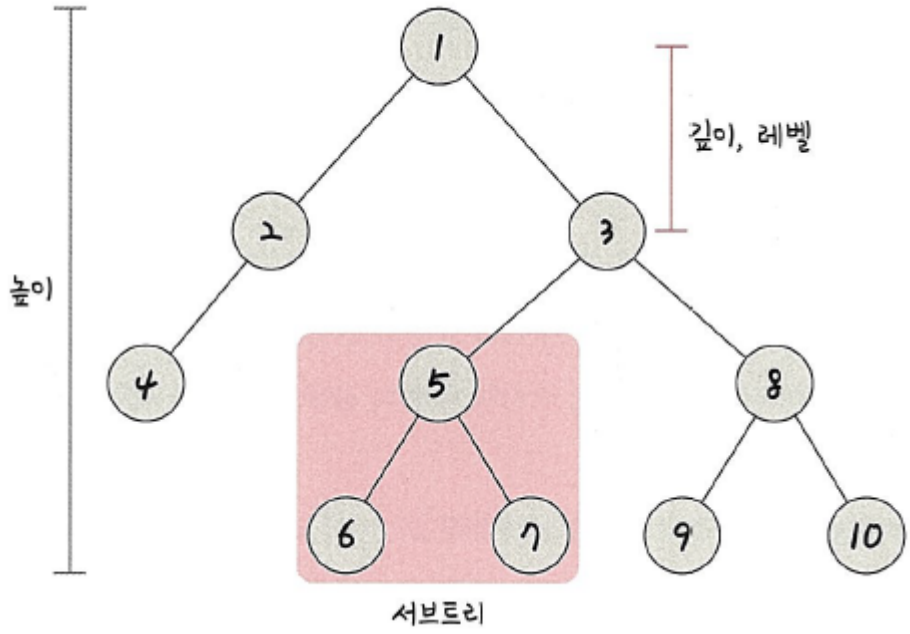
내부 노드

루트 노드와 리프 노드 사이에 있는 노드

리프 노드

자식 노드가 없는 노드

트리의 높이와 레벨



- **깊이**

  각 노드마다 다름. 루트 노드부터 특정 노드까지 최단 거리로 갔을 때의 거리

- **높이**

루트 노드부터 리프 노드까지 거리 중 가장 긴 거리

- **레벨**

  일반적으로 깊이와 같은 의미를 지님

- **서브트리**

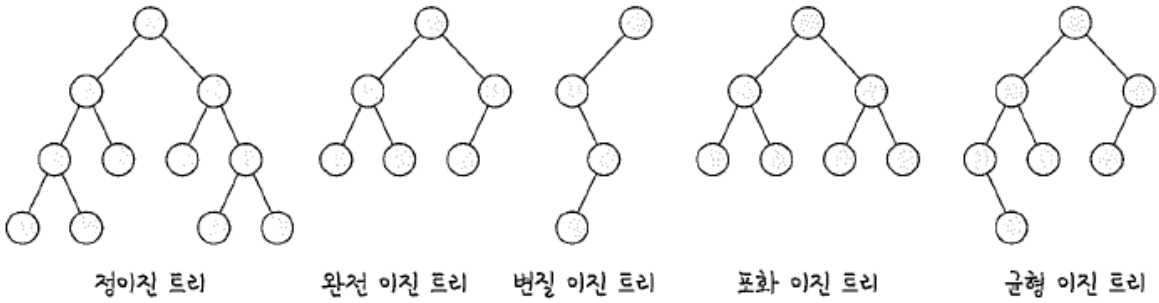
  트리 내의 하위 집합. 하위 트리

  트리 내에 있는 부분집합이라고도 보면됨

이진 트리

자식의 노드 수가 두 개 이하인 트리

이진 트리의 종류



- **정이진 트리(Full Binary Tree)**

  자식 노드가 0 또는 두 개인 이진 트리

- **완전 이진 트리 (Complete Binary Tree)**

  왼쪽에서부터 채워져 있는 이진 트리

  마지막레벨을 제외하고는 모든 레벨이 완전히 채워져 있으며, 마지막 레벨의 경우 왼쪽부터 채워져 있는 형태의 트리

- **변질 이진 트리 (Degenerate Binary Tree)**

  자식 노드가 하나 밖에 없는 이진 트리

- **포화 이진 트리 (Perfect Binary Tree)**

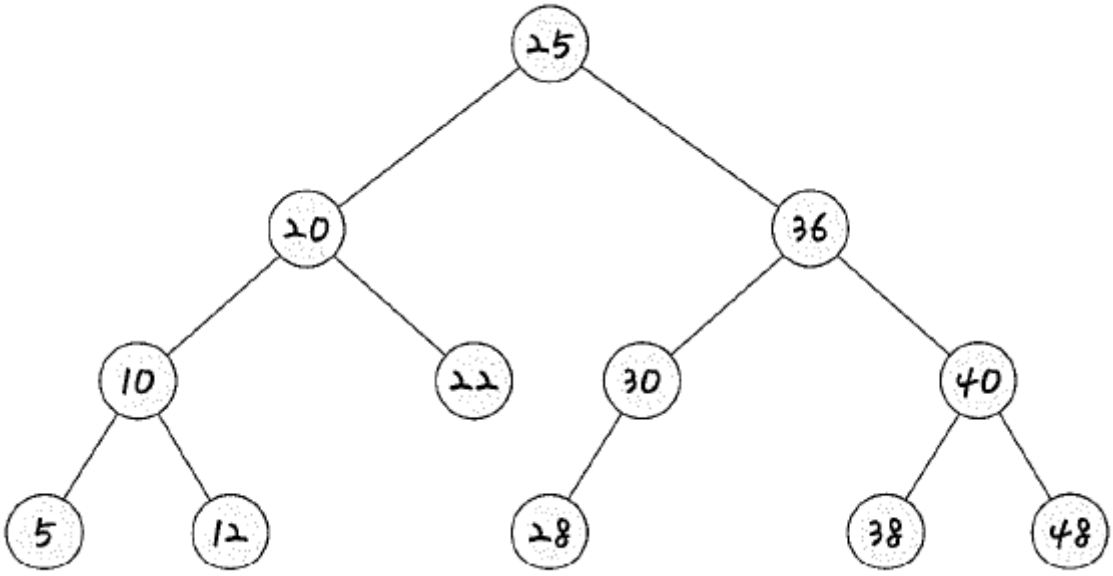
  모든 노드가 꽉 차 있는 이진 트리

- **균형 이진 트리 (Balanced Binary Tree)**

  왼쪽과 오른쪽 노드의 높이 차이가 1 이하인 이진 트리

  map, set 을 구성하는 레드 블랙 트리는 균형 이진 트리 중 하나

이진 탐색 트리(BST)

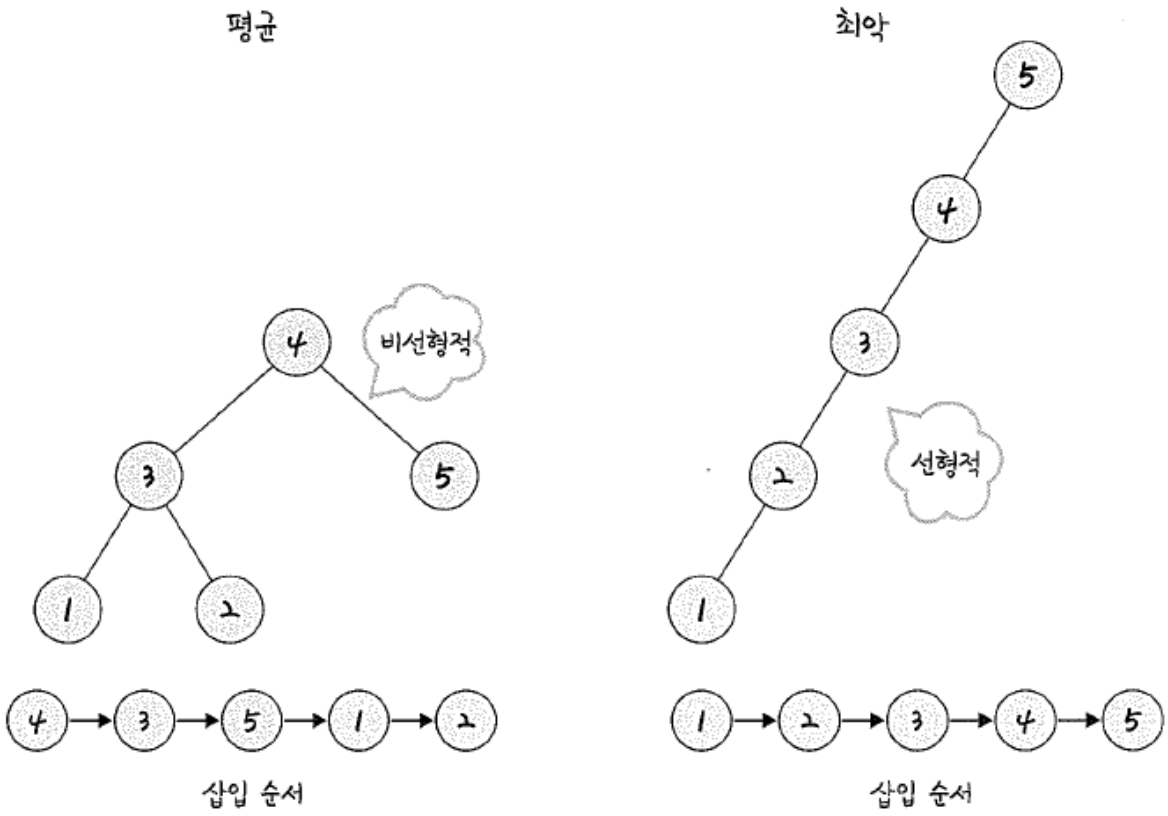


노드의 오른쪽 하위 트리에는 노드 값보다 큰 값이 있는 노드만 포함되고, 왼쪽 하위 트리에는 노드 값보다 작은 값이 들어 있는 트리

왼쪽 및 오른쪽 하위트리도 해당 특성을 가짐

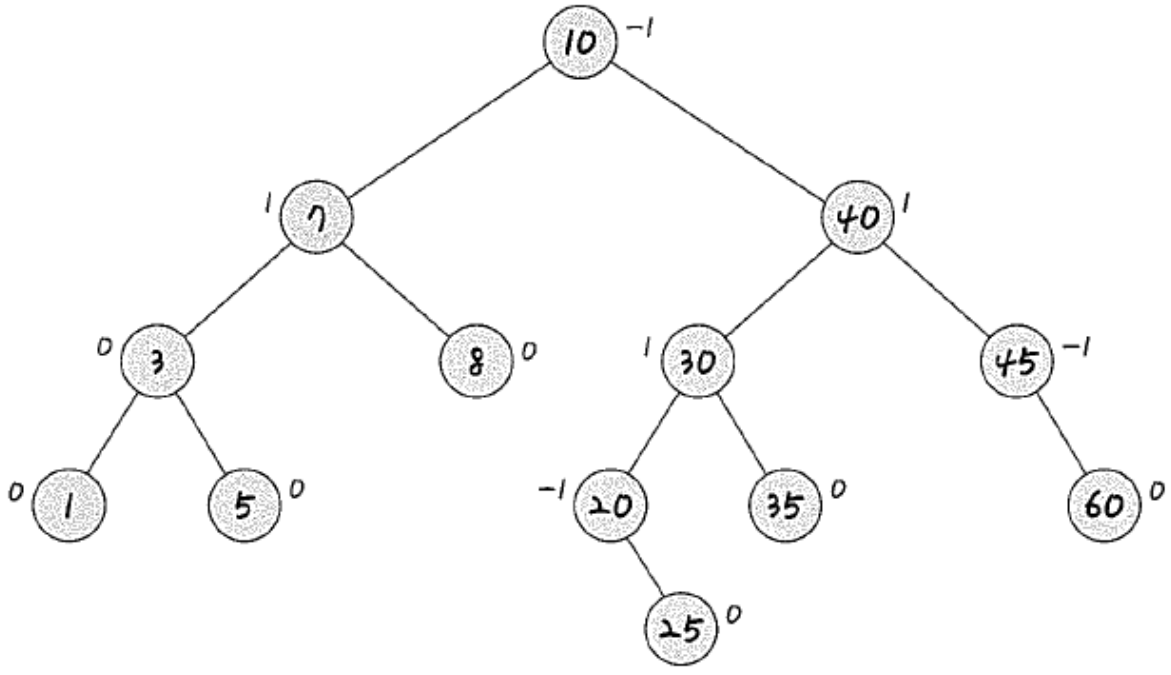
검색에 용이한 구조

보통 $O(logn)$이 소요되나, 최악의 경우는 $O(n)$ 가 소요됨



- 이진 탐색 트리는 삽입 순서에 따라 선형적일 수 있기 때문

AVL 트리(Adelson-Velsky and Landis Tree)



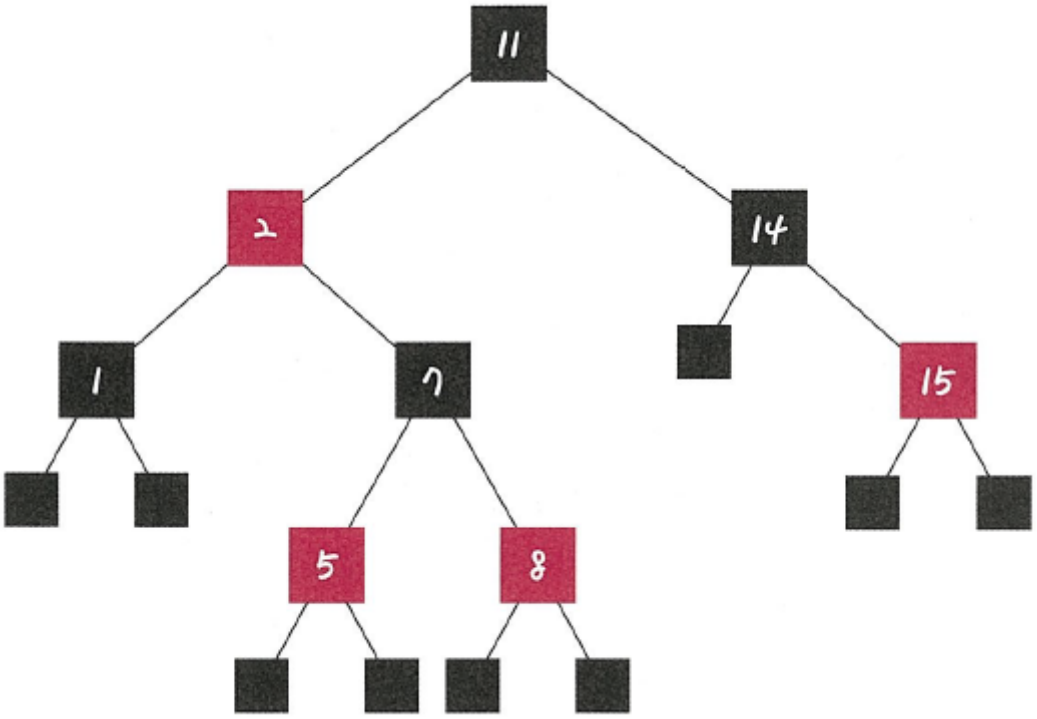
최악의 경우 선형적인 트리가 되는 것을 방지하고 스스로 균형을 잡는 이진 트리

최악의 경우를 배제하고 항상 균형 잡힌 트리로 만든다는 개념을 가진 트리

두 자식 서브트리의 높이는 항상 최대 1만큼 차이가 난다는 특징이 있음

탐색, 삽입, 삭제 모두 시간 복잡도가 $O(logn)$이며 삽입, 삭제 시 마다 균형이 안 맞는 것을 맞추기 위해 트리 일부를 왼쪽 혹은 오른쪽으로 회전시키며 균형을 잡음

레드 블랙 트리



균형 이진 탐색 트리

탐색, 삽입, 삭제 모두 시간 복잡도가 $O(logn)$

각 노드는 빨간색 또는 검은색의 색상을 나타내는추가 비트를 저장함

삽입 및 삭제 중에 트리가 균형을 유지하도록 하는데 사용됨

모든 리프 노드와 루트 노드는 블랙이고 어떤 노드가 레드이면, 그 노드의 자식은 반드시 블랙이다

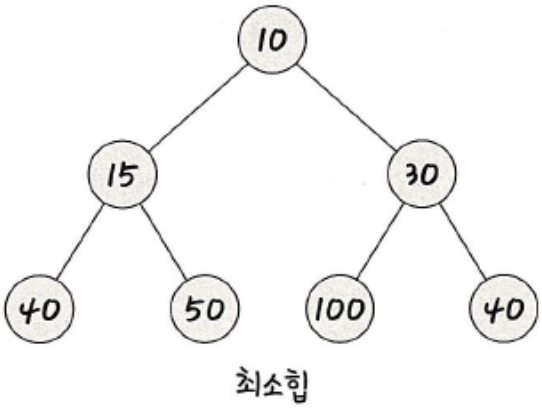
등의 규칙을 기반으로 균형을 잡는 트리

힙(heap)

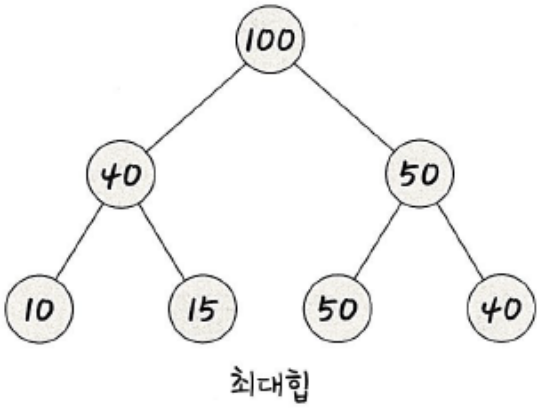
완전 이진 트리 기반의 자료구조

최소힙과 최대힙의 두가지가 잇음

해당 힙에 따라 특정한 특징을 지킨 트리



최소힙



최대힙

최대힙

루트 노드에 있는 키는 모든 자식에 있는 키 중에서 최대여야함

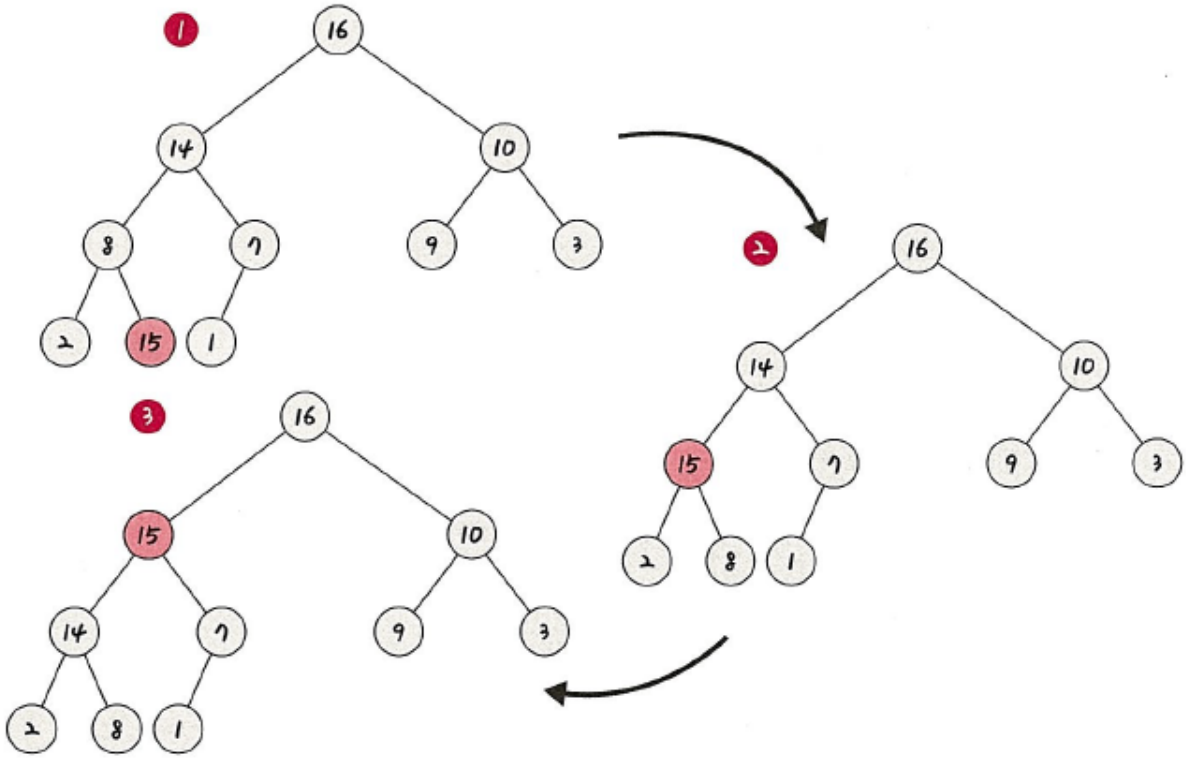
각 노드의 자식 노드와의 관계도 이와 같은 특징이 재귀적으로 이루어져야함

최소힙

루트 노드에 있는 키는 모든 자식에 있는 키 중에서 최소여야함

각 노드의 자식 노드와의 관계도 이와 같은 특징이 재귀적으로 이루어져야함

최대 힙의 삽입



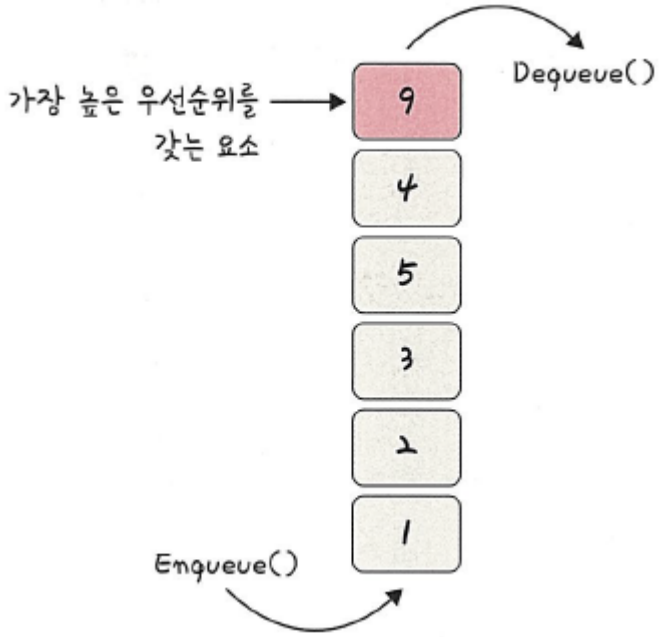
힙에 새로운 요소가 들어오면, 일단 새로운 노드를 힙의 마지막 노드에 이어서 삽입

이 새로운 노드를 부모 노드들과의 크기를 비교하며 교환해서 힙의 성질을 만족시킴

최대 힙의 삭제

최댓값은 루트 노드이므로 루트 노드가 삭제되고, 그 이후 마지막 노드와 루트 노드를 스왑하여 또다시 스왑하는 등의 과정을 거쳐 재구성됨

우선 순위 큐 (Priority Queue)



대기열에서 우선순위가 높은 요소가 우선순위가 낮은 요소보다 먼저 제공되는 자료구조

힙을 기반으로 구현됨

맵 (Map)

특정 순서에 따라 키와 매핑된 값의 조합으로 형성된 자료구조

레드 블랙 트리 자료구조를 기반으로 형성되며, 삽입 시 자동 정렬됨

셋 (Set)

특정 순서에 따라 고유한 요소를 저장하는 컨테이너

중복되는 요소는 없으며 오로지 희소한(unique) 값만 저장하는 자료 구조

해시 테이블

무한에 가까운 데이터들을 유한한 개수의 해시 값으로 매핑한 테이블

삽입, 삭제, 탐색 시 평균적으로 O(1) 의 시간 복잡도를 가짐