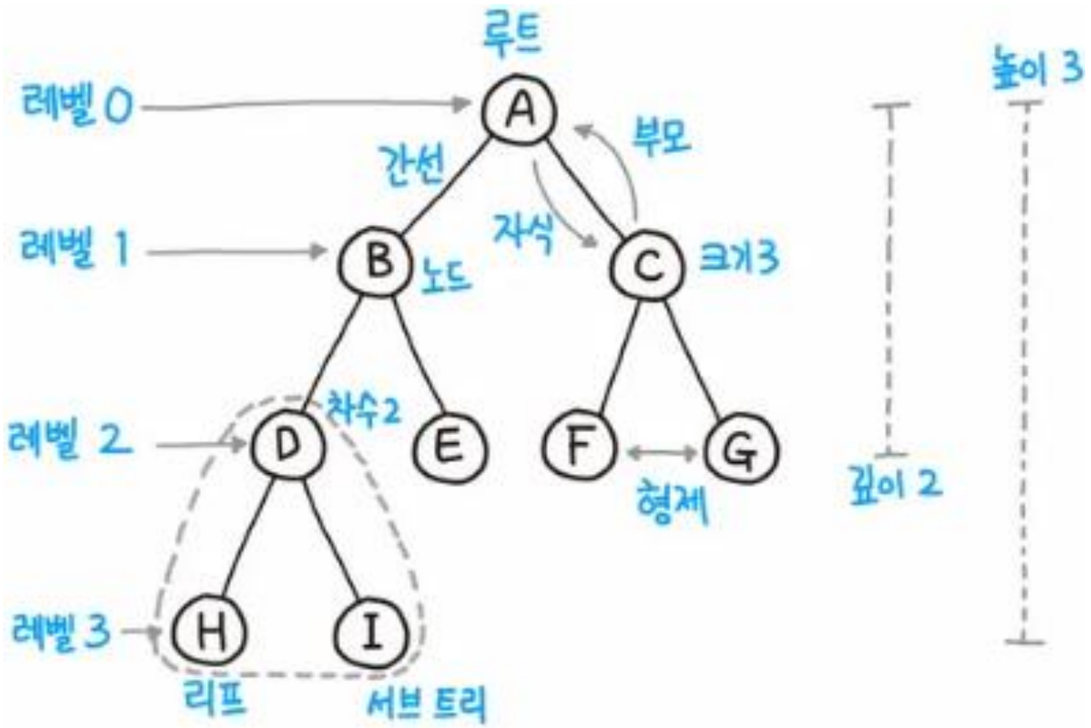
2025 겨울방학 알고리즘 스터디

트리 기초

컴퓨터 공학과 20230546 서보경

목차

트리가 뭘까?



트리 구조는 나무를 반대로 뒤집어 놓은 듯한 형태를 가진다.
마치 나무의 맨 밑이 뿌리로 시작해서 마지막이 잎의 형태를 가지는 것처럼 말이다.

자료구조에서의 트리는 항상 루트에서부터 시작 되며,
부모-자식 관계와 같은 계층적 관계를 가진다.
루트를 제외한 각 노드들은 어떤 노드의 부모 혹은 자식이며
만약 현재 노드에서 자식이 없다면 그 노드를 leaf 노드라고 한다. (루트 노드의 부모는 없으며, 자식만이 존재한다.)

차수 : 자식 노드의 개수
크기 : 자신을 포함한 모든 자식노드의 개수
높이 : 현재 위치에서 부터 리프(leaf) 까지의 거리
깊이 : 루트에서부터 현재 노드까지의 거리

트리는 항상 단방향이고, 사이클이 없는 그래프이다.
일반적으로 방향은 위에서 아래로 향한다.
(그래프 수업에서 다시 다룰 예정)

트리는 왜 쓰는 걸까?

문제 상황 : 데이터가 계속 삽입될 때 특정 값 찾기

배열

트리

트리는 왜 쓰는 걸까?

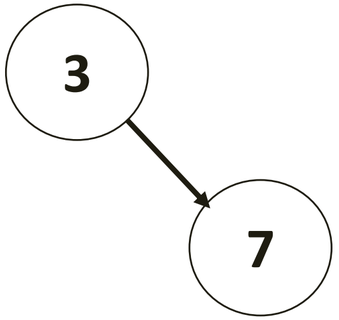
문제 상황 : insert 3 , search 3

배열

트리

트리는 왜 쓰는 걸까?

문제 상황 : insert 7 , search 7



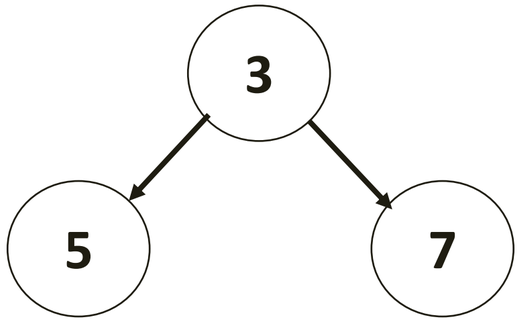
3	7			

배열

트리

트리는 왜 쓰는 걸까?

문제 상황 : insert 5 , search 7



3	7	5		

배열

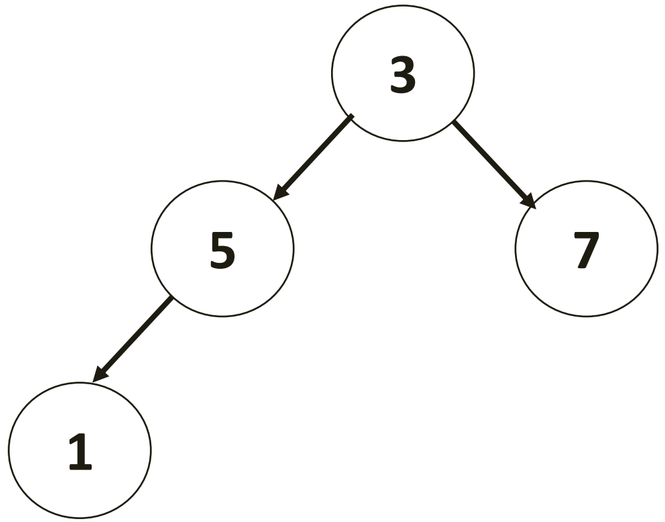
트리

트리는 왜 쓰는 걸까?

문제 상황 : insert 1 , search 1

3	7	5	1	

배열



트리

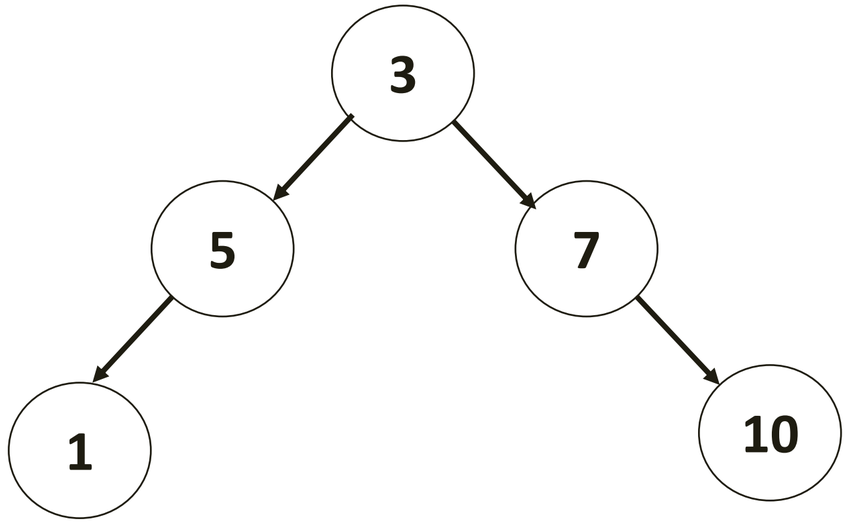
트리는 왜 쓰는 걸까?

문제 상황 : insert 10 , search 1

데이터가 더 방대해 진다면?

3	7	5	1	10

배열



트리

트리는 왜 쓰는 걸까?

특정 데이터에 따라 배열 구조가 이득일수도 있다.
그러나 데이터가 방대하고 삽입이 계속 이루어지는 온라인 상태에서 값을 구해야한다면?
배열을 정렬하며 탐색을 해도 되지만 정렬 코스트가 O(nlogn)인 상황에서 비용이 너무 쎄다!
또한 배열은 크기가 고정 되어 있어 새로운 데이터가 추가되면 메모리를 재할당 해야하는데, 이는 비용이 너무 크다.

특징	트리(Tree)	배열(Array)
구조	계층적 구조(부모-자식 관계)	선형적 구조
삽입 시간 복잡도	O(log n)	O(1) / O(n) (중간 삽입)
삭제 시간 복잡도	O(log n)	O(1) / O(n) (중간 삭제)
탐색 시간 복잡도	O(log n)	O(n)(정렬X) / O(logn)(정렬 후 이분 탐색)
정렬 여부	삽입시 자동 정렬	별도 정렬 작업 필요 (O(n logn))
크기 조정	동적 크기 조정 가능	고정 크기
순차접근	O(log n) / 트리 순회	O(1)
메모리 사용	각 노드에 포인터 저장	데이터만 저장하므로 효율적
요약	삽입,삭제,탐색이 빈번한 경우에 유리하며, 계층적 구조나 정렬된 데이터를 다룰때 유리함.	순차적 데이터 접근을 하거나, 고정 크기 데이터를 다룰 때 유리하다.