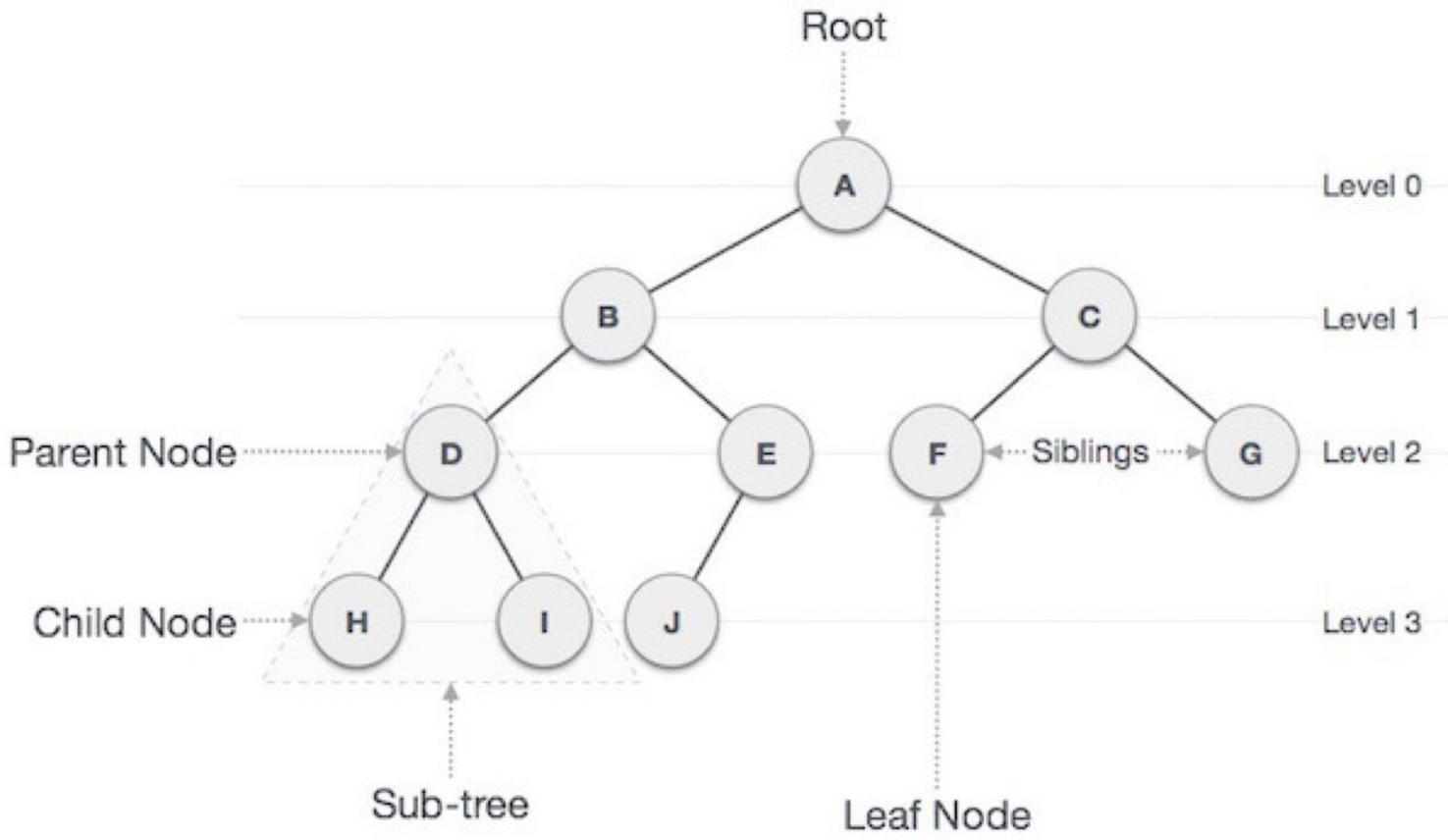
자료 구조 스터디 04

2021-2 KCA

트리와 그래프 + BFS, DFS

본 ppt의 자료는 Pt.J님의 자료와 김성열 교수님의 강의 및 여러 블로그를 참고하였음을 밝힙니다.

트리



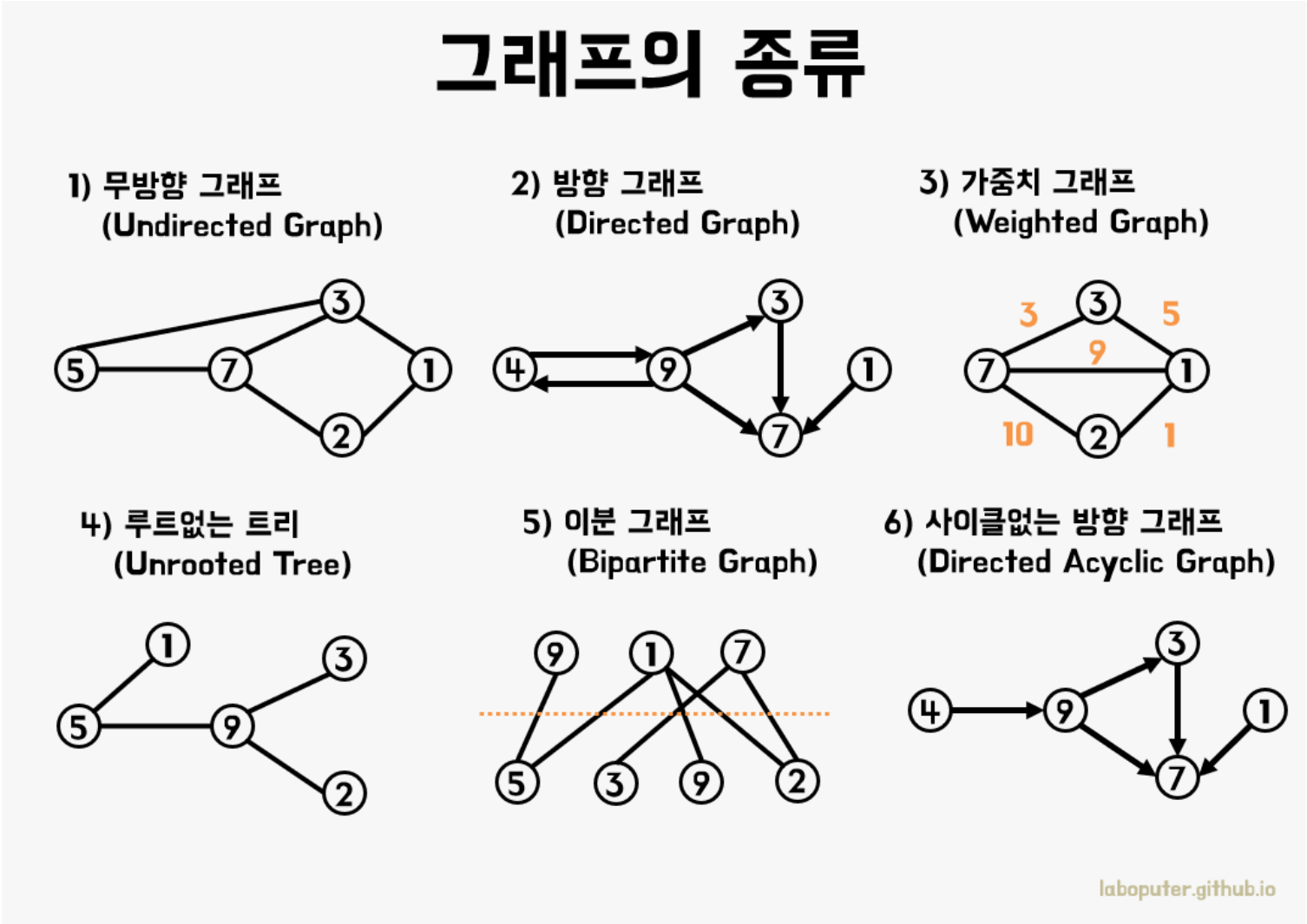
edge(간선) : 노드와 노드를 연결하는 선

depth(해당 노드의 깊이) : 루트 노드에서 해당 노드에 도착하기까지 거쳐야 하는 간선(노드)의 수

height(트리의 높이) : 트리에서 가장 깊이 있는 노드의 depth

https://gmlwjd9405.github.io/2018/08/12/data-structure-tree.html

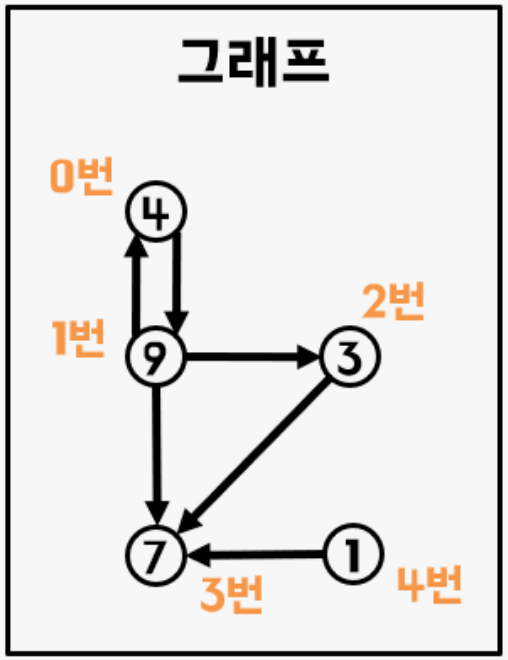
그래프



https://laboputer.github.io/ps/2017/09/29/graph/

그래프의 표현

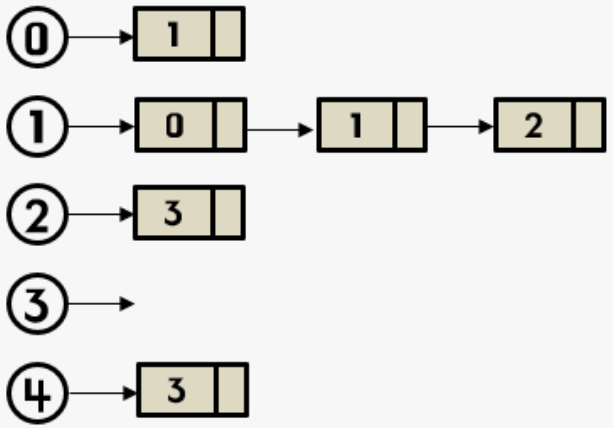
- 인접 행렬(Adjacency Matrix) : O(V^2) 메모리 필요
- 인접 리스트(Adjacency List) : O(V+E) 메모리 필요 → 간선이 적은 경우 유리



1) 인접 행렬 표현

	0	1	2	3	4
0	0	1	0	0	0
1	1	0	1	1	0
2	0	0	0	1	0
3	0	0	0	0	0
4	0	0	0	1	0

2) 인접 리스트 표현



laboputer.github.io

https://laboputer.github.io/ps/2017/09/29/graph/

트리와 그래프

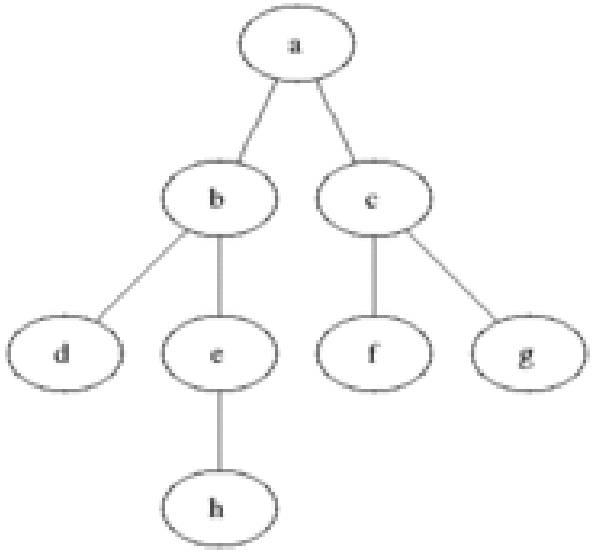
- 트리는 그래프의 일종이다.
- DAG (Directed Acyclic Graph, 방향성을 가지며 사이클이 없는 그래프)의 한 종류가 트리.

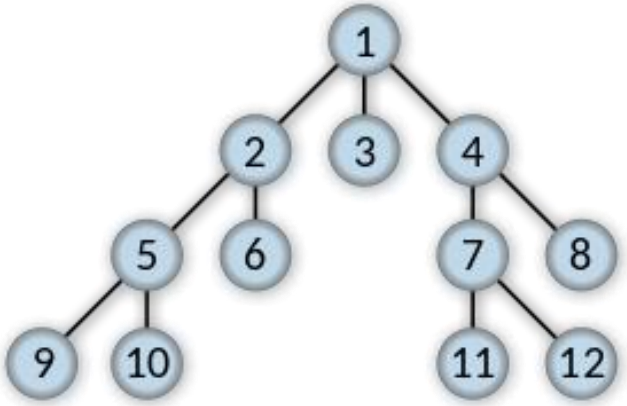
- 트리는 자식이 부모를 하나만 가질 수 있는 DAG 이다.

	그래프	트리
정의	노드(node)와 그 노드를 연결하는 간선(edge)을 하나로 모아 놓은 자료 구조	그래프의 한 종류 DAG(Directed Acyclic Graph, 방향성이 있는 비순환 그래프)의 한 종류
방향성	방향 그래프(Directed), 무방향 그래프(Undirected) 모두 존재	방향 그래프(Directed Graph)
사이클	사이클(Cycle) 가능, 자체 간선(self-loop)도 가능, 순환 그래프(Cyclic), 비순환 그래프(Acyclic) 모두 존재	사이클(Cycle) 불가능, 자체 간선(self-loop)도 불가능, 비순환 그래프(Acyclic Graph)
루트 노드	루트 노드의 개념이 없음	한 개의 루트 노드만이 존재, 모든 자식 노드는 한 개의 부모 노드 만을 가짐
부모-자식	부모-자식의 개념이 없음	부모-자식 관계 top-bottom 또는 bottom-top으로 이루어짐
모델	네트워크 모델	계층 모델
순회	DFS, BFS	DFS, BFS안의 Pre-, In-, Post-order
간선의 수	그래프에 따라 간선의 수가 다름, 간선이 없을 수도 있음	노드가 N인 트리는 항상 N-1의 간선을 가짐
경로	-	임의의 두 노드 간의 경로는 유일
예시 및 종류	지도, 지하철 노선도의 최단 경로, 전기 회로의 소자들, 도로(교차점과 일방 통행길), 선수 과목	이진 트리, 이진 탐색 트리, 균형 트리(AVL 트리, red-black 트리), 이진 힙(최대힙, 최소힙) 등

https://gmlwjd9405.github.io/2018/08/12/data-structure-tree.html

BFS (Breadth First Search) 너비 우선 탐색





하나의 노드를 방문한 후,
그 노드에 바로 인접해 있는 모든 노드를 먼저 방문하는 방법

큐를 이용해 구현한다.

시작 노드에서 목표 노드까지 최단 길이 경로가 보장된다.
(== 다익스트라에서 모든 간선의 가중치가 1이면 BFS 이다.)

https://en.wikipedia.org/wiki/Breadth-first_search

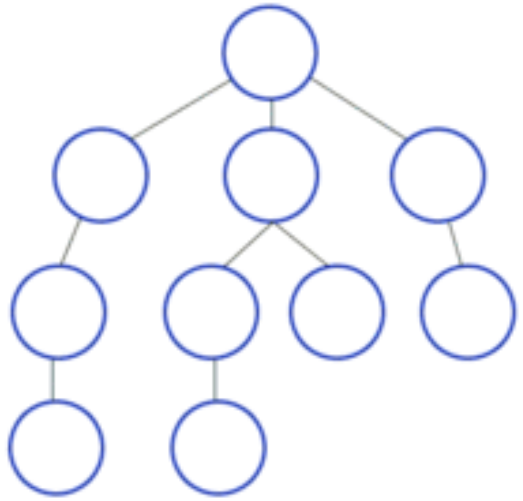
BFS 구현

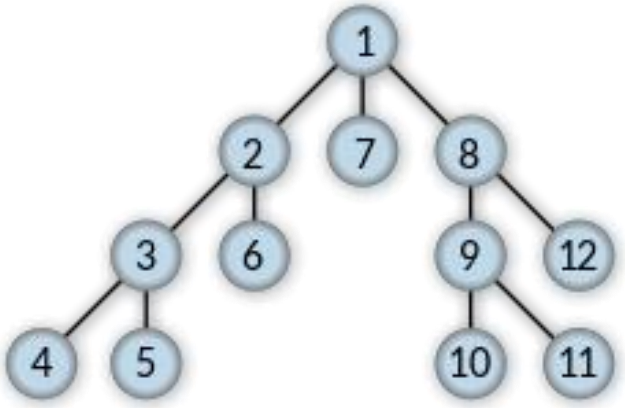
- 큐를 이용
- root 노드를 넣은 상태로 시작.
- 큐에서 노드를 pop하고, 그 노드의 인접 노드를 모두 큐에 push

=> 이를 반복.

<더 자세한 설명은 여기로..>
D8_Search 13p~29p (pt.j)
https://blog.encrypted.gg/941?category=773649

DFS (Depth First Search) 깊이 우선 탐색





하나의 노드를 방문한 후,
그 노드에 인접한 노드 중 하나를 선택해서 최대 깊이(depth)에 도달할 때까지 계속 탐색하는 방법.

스택을 이용해 구현한다.

탐색 과정이 무한히 진행되는 걸 막기 위해, 깊이 제한을 둔다.

깊이 제한에 도달해서 다시 부모 노드를 타고 돌아오는 과정을 “백트래킹” 이라고 한다.

https://en.wikipedia.org/wiki/Depth-first_search

DFS 구현

- 스택을 이용
- root 노드를 넣은 상태로 시작.
- 스택에서 노드를 확인
  - 그 노드의 인접 노드가 있다면, 인접 노드 중 하나를 스택에 push
  - 그 노드의 인접 노드가 없다면, pop
- 이를 반복

<더 자세한 설명은 여기로..>
D8_Search 30p~ (pt.j)
https://blog.encrypted.gg/942?category=773649

문제를 풀어나갈 때

- 주어진 문제 상황을 어떻게 그래프, 트리로 표현할 수 있을까?

- 그래프, 트리를 어떻게 메모리에 집어넣을 수 있을까?
  => 배열, 연결리스트

- 그래프, 트리를 어떻게 탐색할 수 있을까?
  => BFS, DFS + 큐, 스택 이용

구현 과제

각자 더미 Tree 생성 후,
BFS, DFS로 트리 순회해서,
모든 노드 값 콘솔에 출력해 보기!

<예시 출력>

```
--root -- 1 -- 11
                -- 111
          -- 2 -- 12 -- 3 -- 4
                              -- 5
```