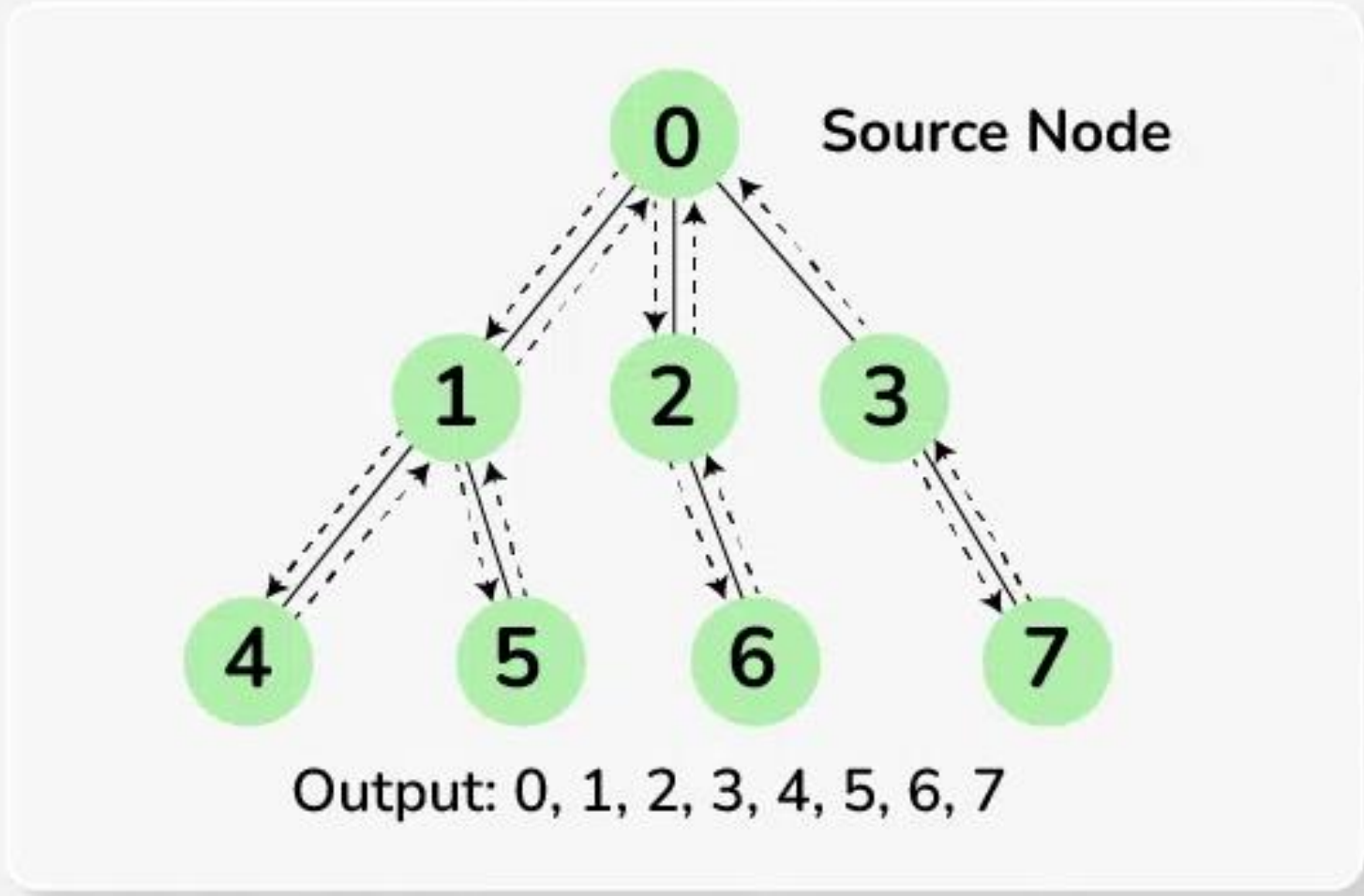
코테스터디

DFS(Depth-first search)
깊이 우선 탐색

Depth First Search



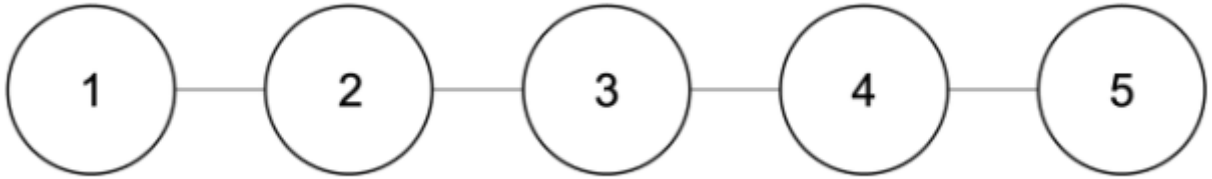
그래프를 탐색 하는 방법 중 하나!


그게 뭔데
씹덕아

선형 구조

선형 구조 Linear 란?

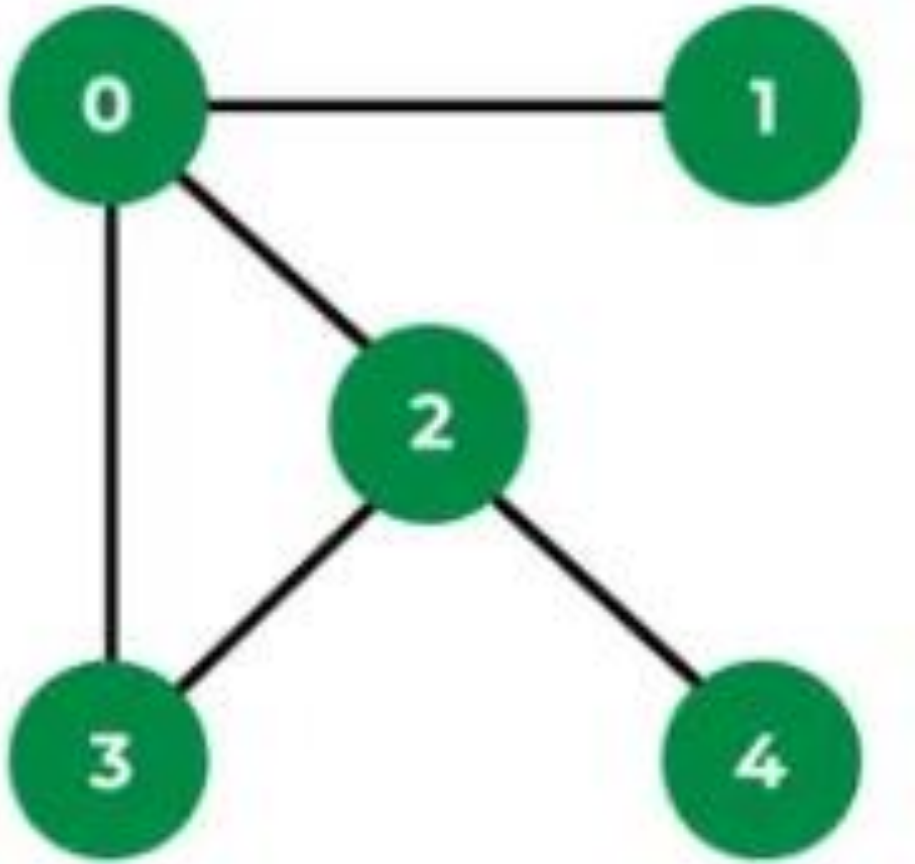
- 자료를 구성하는 **원소들을 하나씩 순차적으로 나열한 상태**
- **자료들간의 앞, 뒤 관계가 1:1관계**로 **배열과 리스트가 대표적**이며 **스택과 큐**도 이에 해당

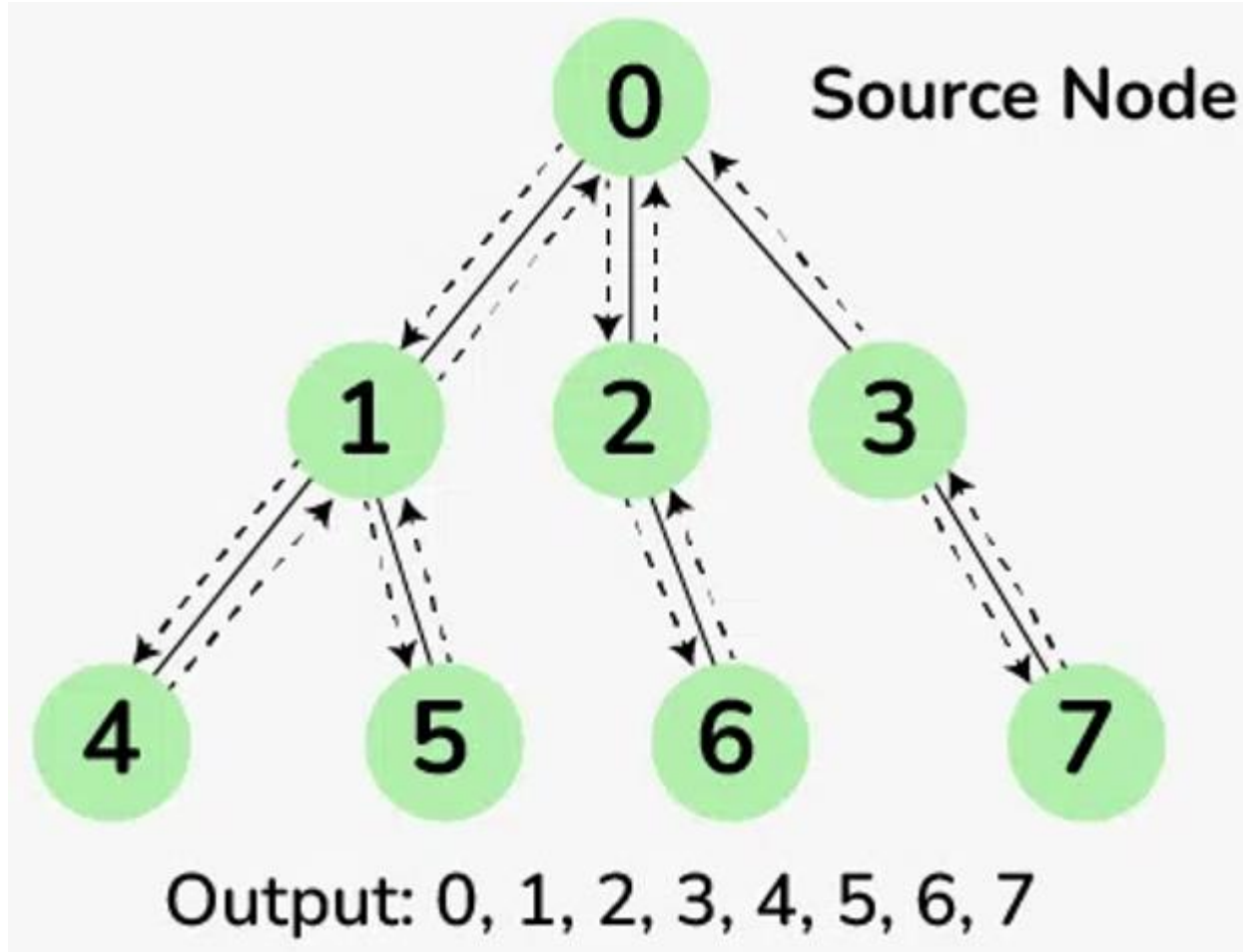


비선형 구조

비선형 구조 Non Linear 란?

- 데이터를 저장하기 위한 방법으로 **데이터 간의 관계**를 이루면서 '**계층적인 구조**'를 가지며 '**일렬로 나열되지 않은 자료구조**' 형태를 의미합니다.
- 일련 되지 않은 자료구조는 계층적으로 **데이터의 관계**가 **부모-자식 관계**, **연결 관계**, 또는 **소속 관계** 등을 가지고 있어서 계층적이거나 상호 연결되어 있습니다.
- 대표적인 비선형 구조는 **트리(Tree)**, **그래프(Graph)**등이 이에 해당합니다.
- **하나의 자료 뒤에 여러 개의 자료가 존재**할 수 있는 형태(1:N)


0
1
2
3
4


0
Source Node
1
2
3
4
5
6
7
Output: 0, 1, 2, 3, 4, 5, 6, 7

그래프 Graph 란?

- 비선형 구조 중 하나로 '**정점(Vertices)**과 그들 사이를 연결하는 **간선(Edge)**으로 표현된 **자료구조**'를 의미합니다. 각 **정점은 데이터를 저장**하며 **간선은 정점들 간의 관계**를 나타냅니다.
- 모든 그래프는 공식적으로 **정점(V)과 간선(E)으로 그래프는 G = {V, E}**로 표현이 됩니다.
- 그래프는 다양한 형태와 용도를 가지며 네트워크, 경로 탐색, 스케줄링 등 다양한 시스템에서 활용됩니다.
- 또한 다양한 알고리즘과 연산을 수행하는데 이용됩니다. 예를 들어 **그래프 탐색 알고리즘을 사용**하여 **특정 노드를 찾거나 최단 경로 알고리즘을 사용**하여 **두 노드 사이의 최단 경로**를 찾을 수 있습니다

트리 Tree 란?

```
class Node {
  public String name;
  public Node[] children;
}
```

- 트리는 하나의 **루트 노드**를 갖는다.
- **루트 노드**는 **0개 이상의 자식 노드**를 갖고 있다.
- 그 자식 노드 또한 0개 이상의 자식 노드를 갖고 있고, 이는 반복적으로 정의된다.
- **노드(node)들과 노드들을 연결하는 간선(edge)**들로 구성되어 있다
- 트리에는 사이클(cycle)이 존재할 수 없다.
- 노드들은 특정 순서로 나열될 수도 있고 그럴 수 없을 수도 있다.
- 각 **노드는 부모 노드로의 연결이 있을 수도 있고 없을 수도 있다**.
- 각 노드는 어떤 자료형으로도 표현 가능하다

트리 Tree 의 키워드

- **루트 노드(root node)**: 부모가 없는 노드, 트리는 하나의 루트 노드만을 가진다.
- **단말 노드(leaf node)**: 자식이 없는 노드, '말단 노드' 또는 '잎 노드'라고도 부른다.
- **내부(internal) 노드**: 단말 노드가 아닌 노드
- **간선(edge):** 노드를 연결하는 선 (link, branch 라고도 부름)
- **형제(sibling):** 같은 부모를 가지는 노드
- **노드의 크기(size):** 자신을 포함한 모든 자손 노드의 개수
- **노드의 깊이(depth):** 루트에서 어떤 노드에 도달하기 위해 거쳐야 하는 간선의 수
- **노드의 레벨(level):** 트리의 특정 깊이를 가지는 노드의 집합
- **노드의 차수(degree):** 하위 트리 개수 / 간선 수 (degree) = 각 노드가 지닌 가지의 수
- **트리의 차수(degree of tree):** 트리의 최대 차수
- **트리의 높이(height):** 루트 노드에서 가장 깊숙히 있는 노드의 깊이

트리를

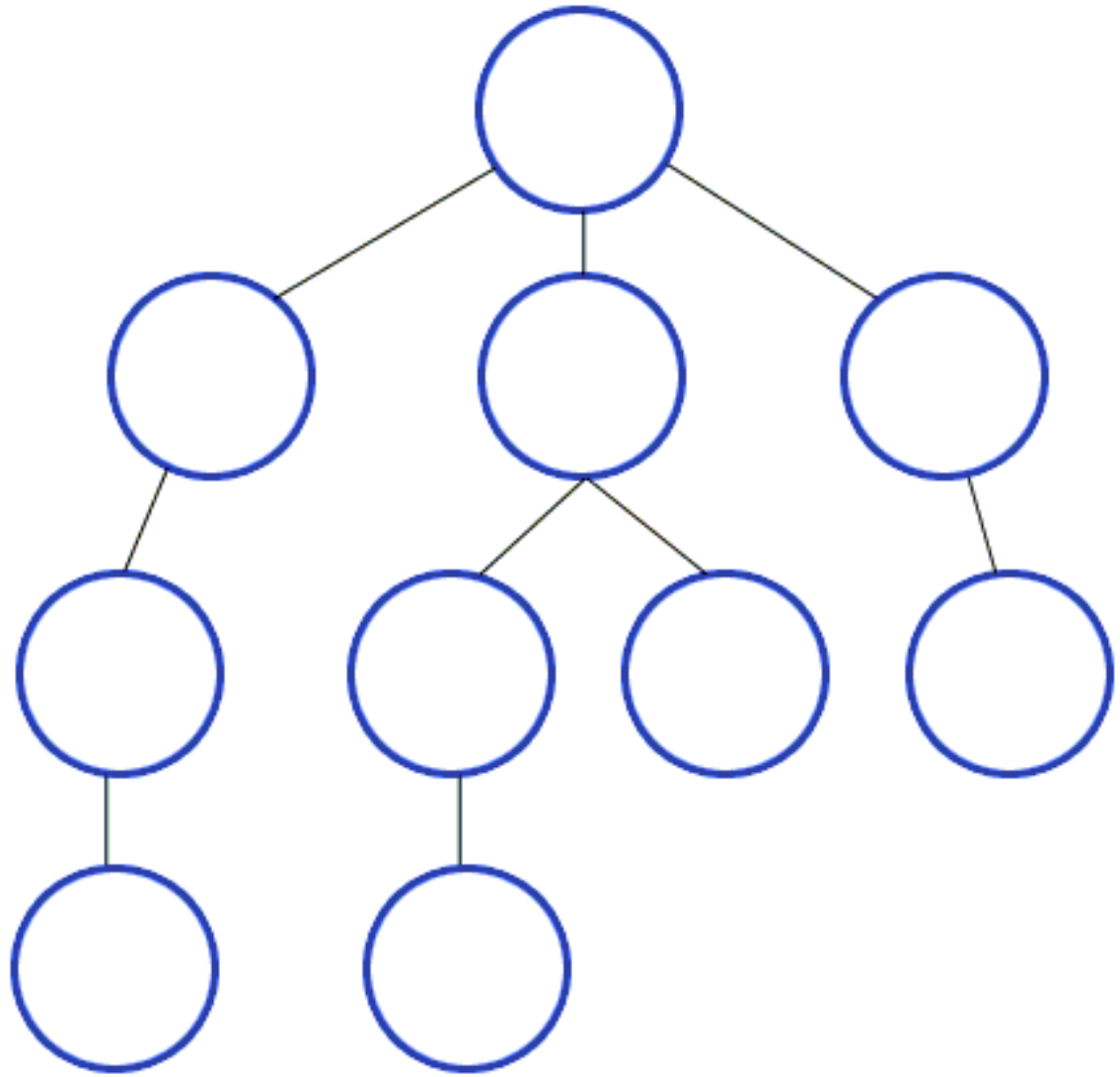
다 설명하기엔..

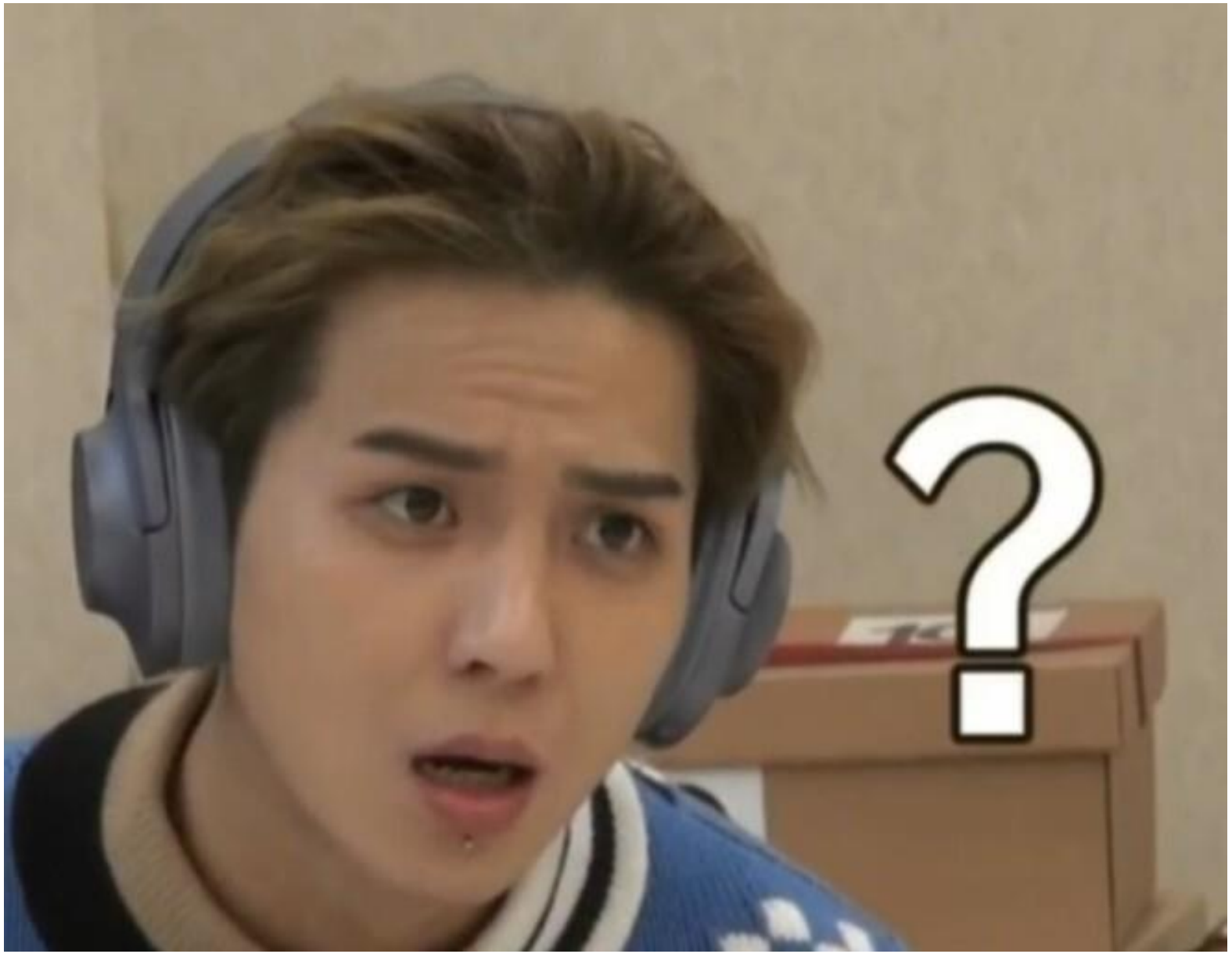

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
오늘 즐거웠어요!
저도요 다음에 또 만나요!

DFS(Depth-first search)
깊이 우선 탐색

DFS 란?

- **그래프와 트리의 깊은 부분을 우선적으로 탐색**하는 알고리즘
- 한 방향으로 갈 수 있을 때까지 최대한 깊게 탐색한 후 **더 이상 갈 수 없게 되면, 다시 돌아와 다음 경로를 탐색**하는 방식을 의미합니다.
- 기본적인 수행과정은 한 노드에서 시작하여 가능한 한 깊숙이 **탐색한 후, 다음 분기**로 넘어갑니다. 그리고 더 이상 탐색할 수 없는 상태에 도달하면, 이전으로 돌아가서 다른 가능한 분기를 탐색합니다.
- '**모든 정점을 방문**' 하고 **연결된 모든 간선을 검사**합니다.
- => 그래프의 **모든 구성 요소를 탐색(완전 탐색)하거나 특정 조건을 만족하는 경로를 찾을 때 유용**합니다.





DFS의 구현

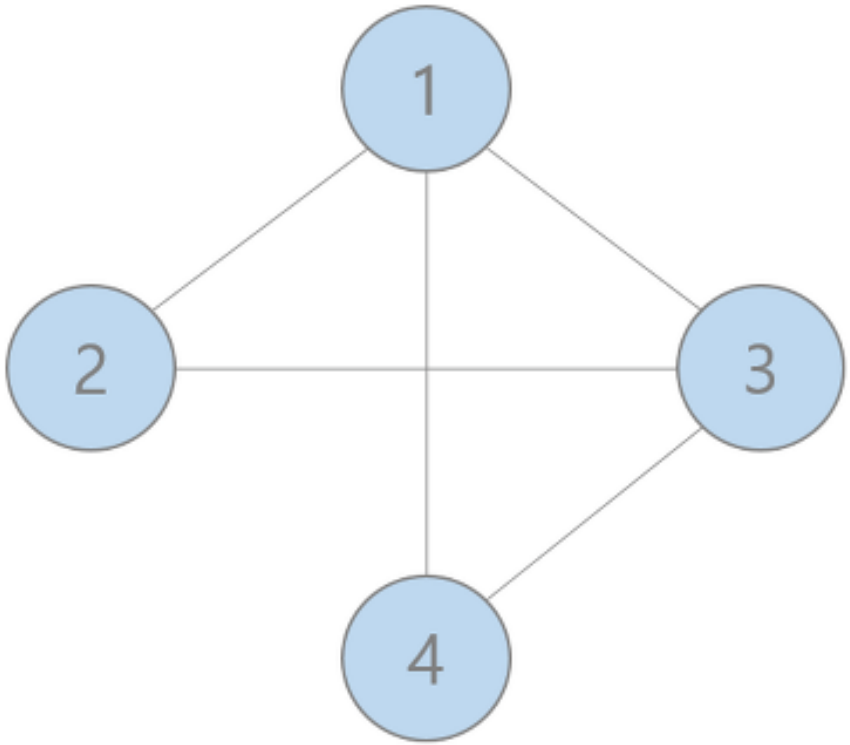
DFS의 구현 – 관계 구현(인접행렬)

인접 행렬은 그래프의 연결 관계를 **이차원 배열**로 나타내는 방식입니다. 인접 행렬을 adj[][][1]라고 한다면 adj[i][j]에 대해서 다음과 같이 정의할 수 있습니다.

> **adj[i][j]** : 노드 i에서 노드 j로 가는 간선이 있으면 1, 아니면 0

cf) 만약 간선에 가중치가 있는 그래프라면 1 대신에 가중치의 값을 직접 넣어주는 방식으로 구현할 수 있습니다.

DFS의 구현 – 관계 구현(인접행렬)



0	1	1	1
1	0	1	0
1	1	0	1
1	0	1	0

DFS의 구현 – 관계 구현

```java
static int[][] graph;
```

```java
static boolean[] visited;
```

```java
for (int i = 1; i <= N; i++) {
    graph[i][i] = 1;
}
```

```java
for (int i = 0; i < M; i++) {
    st = new StringTokenizer(br.readLine());
    int a = Integer.parseInt(st.nextToken());
    int b = Integer.parseInt(st.nextToken());
    graph[a][b] = 1;
    graph[b][a] = 1;
}
```

Graph = 관계를 표시할 저장소

Visited = 방문할 노드를 표시할 Boolean 배열

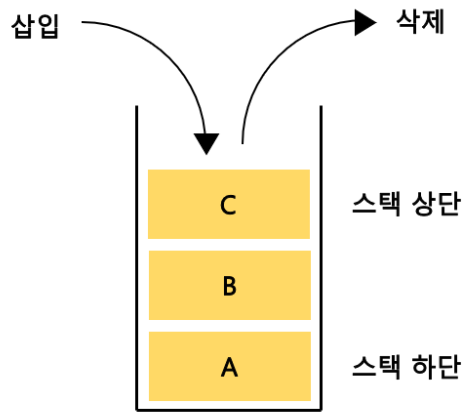
본인과 본인은 연결되어있으므로 초기화

선이 방향을 아닌 양방향이므로 서로 연결해준다.

DFS의 구현 - 재귀

```java
public static void dfs(int start) {  2 usages
    if(!visited[start]) {
        visited[start] = true;
    }

    for (int i = 1; i <= N; i++) {
        if (graph[start][i] == 1 && !visited[i]) {
            dfs(i);
        }
    }
```



DFS의 구현 - 스택

```java
public static void dfs2(int start) {  1 usage
    Stack<Integer> stack = new Stack<>();
    stack.push(start);

    while (!stack.isEmpty()) {
        int top = stack.pop();
        if(!visited[top]){
            visited[top] = true;
            for (int i = N; i >= 1; i--) {
                if (graph[top][i] == 1 && !visited[i]) {
                    stack.push(i);
                }
            }
        }
    }
}
```


뭐라는
거야...
7


어림반푼어치도 없는 애기죠!


YouTube
그런데, 짜잔!

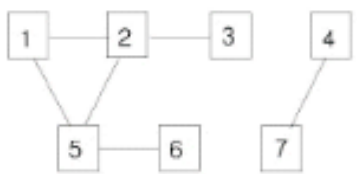
바이러스

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	128 MB	180562	84686	56509	46.001%

문제

신종 바이러스인 웜 바이러스는 네트워크를 통해 전파된다. 한 컴퓨터가 웜 바이러스에 걸리면 그 컴퓨터와 네트워크 상에서 연결되어 있는 모든 컴퓨터는 웜 바이러스에 걸리게 된다.

예를 들어 7대의 컴퓨터가 <그림 1>과 같이 네트워크 상에서 연결되어 있다고 하자. 1번 컴퓨터가 웜 바이러스에 걸리면 웜 바이러스는 2번과 5번 컴퓨터를 거쳐 3번과 6번 컴퓨터까지 전파되어 2, 3, 5, 6 네 대의 컴퓨터는 웜 바이러스에 걸리게 된다. 하지만 4번과 7번 컴퓨터는 1번 컴퓨터와 네트워크상에서 연결되어 있지 않기 때문에 영향을 받지 않는다.



< 그림 1 >

어느 날 1번 컴퓨터가 웜 바이러스에 걸렸다. 컴퓨터의 수와 네트워크 상에서 서로 연결되어 있는 정보가 주어질 때, 1번 컴퓨터를 통해 웜 바이러스에 걸리게 되는 컴퓨터의 수를 출력하는 프로그램을 작성하시오.

DFS 의 장단점

장점

- 현재 경로상의 노드들만 기억하면 되므로 저장 공간의 수요가 비교적 적음
- 목표 노드가 깊은 단계에 있는 경우 해를 빨리 구할 수 있음
- BFS 보다 **구현이 간단**함

단점

- 단순 검색 속도가 **BFS 보다 느림**
- 해가 없는 경우에 빠질 수 있음. 따라서 **사전에 탐색할 임의의 깊이를 지정할 필요**가 있다.
- **DFS는 해를 구하면 탐색이 종료**된다. 따라서 구한 해가 **최적의 해가 아닐 수 있다.**


다시
오늘하루고생했다