ALOHA
#3주차

DFS, BFS, Backtracking

BFS와 거리 계산

그래프에서 최단거리를 계산할 때, 주로 BFS를 사용한다.

-> 각 노드를 방문했다는 표시 = 거리로 설정하여 출력만 하면 끝이기 때문!

오른쪽 예시에서 dist 벡터는 BFS로 방문했는지 확인함과 동시에,
시작점으로부터의 거리를 저장한다!

dist[x] == 0이면 x노드를 방문하지 않은 것이고,
dist[x] != 0이면 시작점으로부터 x노드까지의 거리가 dist[x]이다.

```cpp
vector<vector<int>> arr;
vector<int> dist;

void BFS(int st)
{
    queue<int> q;

    q.push(st);
    while (q.size())
    {
        int x = q.front(); q.pop();
        for (int i = 0; i < arr[i].size(); i++)
        {
            if (not_visited(i))
            {
                q.push(i);
                dist[i] = dist[x]+1;
            }
        }
    }
}
```

인접행렬 vs 인접리스트

인접행렬과 (std::vector로 구현한)인접리스트는 각각 언제 사용하는 것이 좋을까?

-> 간선의 개수가 많다면 인접행렬로, 적다면 인접리스트를 사용하는 것이 좋다!

간선의 개수가 적은데 인접행렬을 사용하면, 0이 저장된 공간이 많아지므로 메모리를 많이 사용한다!

재귀함수를 호출하는 타이밍

DFS에서 스택이 아니라 재귀함수를 이용하여 DFS를 작성한다면 조심해야 할 것이 있다!

"현재 노드를 방문했다고 표시한 다음, 재귀함수를 호출할 것!"

노드 A와 B가 연결되어 있고 A에서 B로 이동하려 할 때, 재귀함수를 먼저 호출해버렸다면?

```
vector<vector<int>> arr;
vector<bool> visited;

void DFS(int node)
{
    for (int i = 0; i < arr[node].size(); i++)
    {
        if (!visited[i])
        {
            DFS(i);
            visited[i] = true;
        }
    }
}
```

A -> B -> A -> B -> A … 무한반복!

함수를 먼저 호출해버려서 현재 노드를 방문했다고 표시하지 못하게 됨!

퀴즈

퀴즈 1

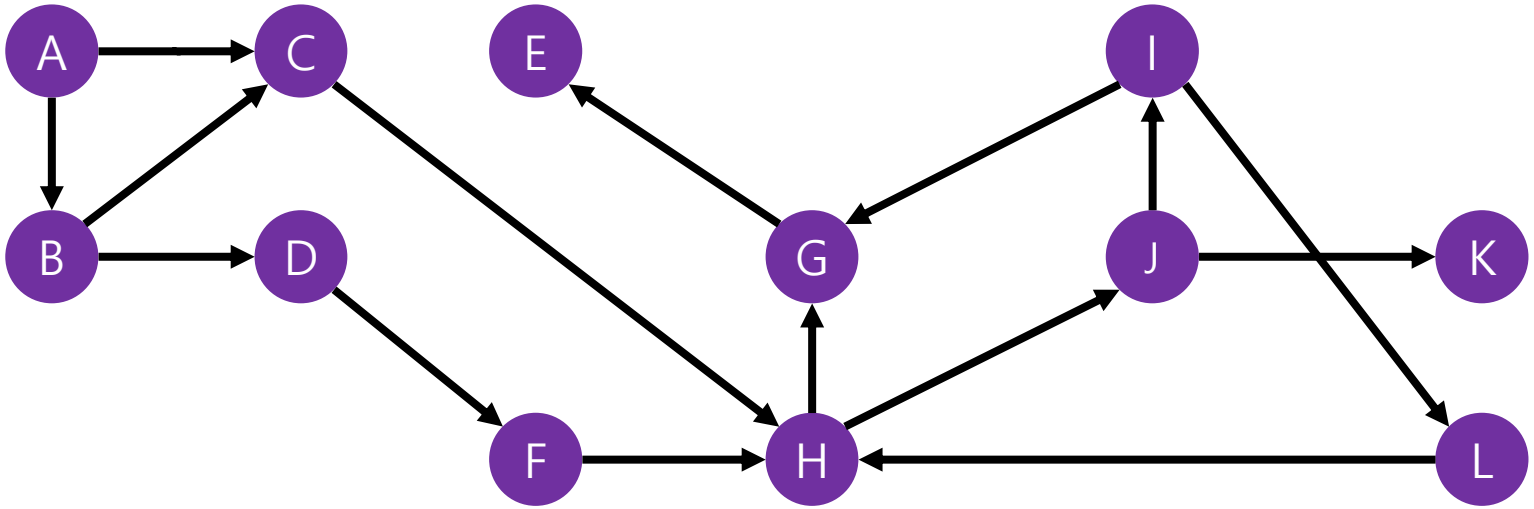
다음 보기 중에서, 알맞은 답을 고르시오.

"BFS는 다음에 방문할 노드를 스택/큐에 저장하고, DFS는 다음에 방문할 노드를 스택/큐에 저장한다."

"BFS/DFS는 재귀함수를 이용하여 구현할 수도 있다."

퀴즈 2

다음 그래프를 “인접행렬”과 “벡터로 구현된 인접리스트”로 표현하시오.

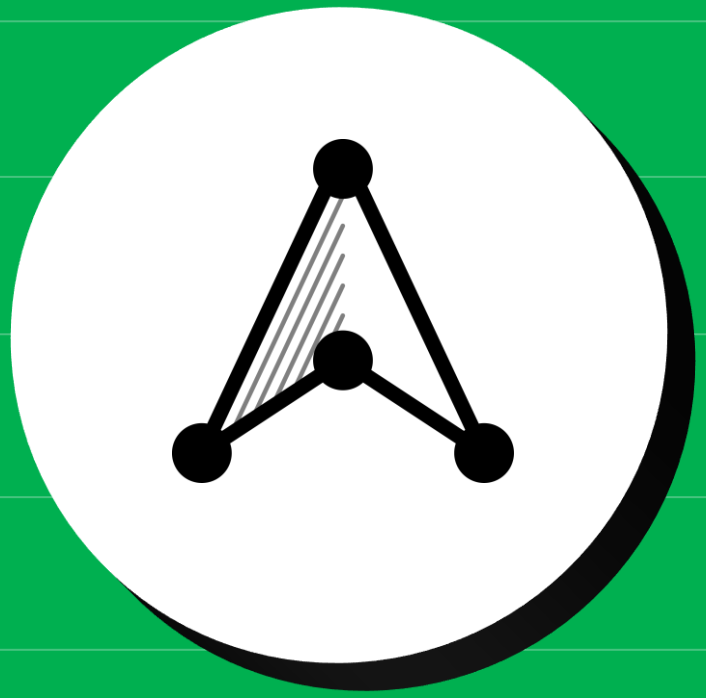


퀴즈 답

1번: 큐, 스택, DFS
2번:

A	B	C
B	C	D
C	H	
D	F	
E		
F	H	
G	E	
H	G	J
I	G	L
J	I	K
K		
L	H	

	A	B	C	D	E	F	G	H	I	J	K	L
A		1	1									
B			1	1								
C								1				
D						1						
E												
F								1				
G					1							
H							1			1		
I							1					1
J									1		1	
K												
L								1				



다음 시간에 만나요~