< 그래프 >

그래프는 여러 개체들이 정점과 간선으로 서로 연결된 자료 구조를 말한다. 그래프를 구현하기위해서는 크게 3가지 방법이 있다.

<그래프를 구현하는 3가지 방법>

1. 인접 행렬

* 인접 행렬은 각 정점에서 다른 정점으로 가는 길이 있는지 없는지를 나아내는 행과 열이다. 정점이 5개 있다면 5X5 인접행렬을 만들어 각 행에서 각 열로 가는 길이 있는지 없는지 정보를 나타냅니다.

1. 인접 리스트

* 인접 리스트는 각 정점이 어떤 정점과 연결되어 있는지 연결리스트로 나타내는 방법입니다.

1. ​격자무늬 그래프

* 격자무늬 그래프는 위에서 올려다본 하나의 미로 판처럼 행과 열 테이블의 한 칸 한 칸을 정점으로 보고 간선은 위, 아래, 왼쪽, 오른쪽(혹은 대각선까지)으로 이동할 수 있다고 가정하는 그래프입니다.

그림, 원, 아동 미술, 도표이(가) 표시된 사진

자동 생성된 설명

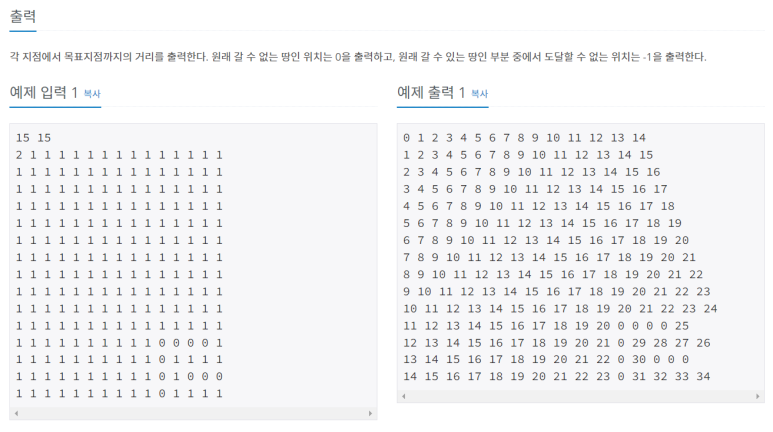
<깊이 우선 탐색(DFS=Depth First Search)>

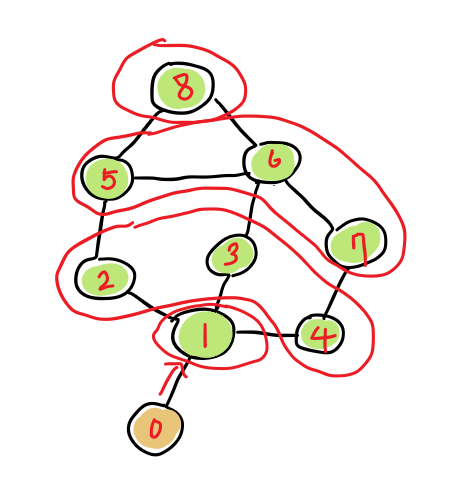
* 깊이 우선 탐색 이론은 기본적으로 스택의 원리에 의해 작동합니다. 내가 탐색한 정점을 기준으로 어떤 간선(edge)을 찾으면 그 간선을 따라 다른 정점으로 이동하고 또 그 정점에서 다른 가선이 있으면 이동하는 형식으로 방문한 정점 정보가 스택에 계속 쌓이다가 (이 때 이미 거쳐온 정점은 다시 가면 안됨) 더 이상 갈 길이 없는 정점이 나오면 스택을 pop하고 pop된 정점에서 또 다른 길이 있는지 찾습니다. 보통 재귀(recursion) 함수를 사용하면 기본적으로 시스템 스택으로 구현되기 때문에 DFS는 흔히 재귀로 구현합니다.
* 위 DFS 그림의 경우 내가 어떤 그래프를 탐색하기 위해 어떤 임의의 조건에 따라 이동하는 DFS 탐색 순서를 나타낸 것입니다. 한 정점에서 두 갈림길 중 어디를 먼저 갈지는 어떻게 설정하느냐에 따라 달라집니다. DFS 의 뜻 그대로 한 노드 한 노드 깊게 간다고 보면 됩니다. 그러다가 갈 곳이 없으면 6에서 7로 가듯이 스택을 pop해서 최신 노드의 다른 갈림길로 가게 된다. 이때 이미 방문한 정점은 또 가지 않는다는 것도 유의해야 합니다, 보통 이미 방분한 정점인 것을 알기 위해 visited(정점의 개수)라는 배열을 따로 만들어 둡니다.
* DFS를 활용하는 알고리즘은 다음과 같다

1. 특정 지점 주변 상 하 좌 우를 살펴본 뒤에 주변 지점 중에서 값이 0 이면서 아직 방문하지 않은 지점이 있다면 해당 지점을 방문한다.
2. 방문한 지점에서 다시 상하좌우를 살펴보면서 방문을 진행하는 과정을 반복하면 연결된 모든 지점을 방문할 수 있다.
3. 모든 노드에 대하여 1~2번의 과정을 반복하며 방문하지 않은 지점의 수를 카운트한다.

<너비 우선 탐색(BFS=Breadth First Search>

* 너비 우선 탐색은 탐색을 시작한 정점에서 점차 멀어져 가며 다른 정점을 탐색합니다. 이를 위해서 큐 자료구조를 사용합니다. BFS 정점의 간선 간에 가중치가 없거나 모두 같을 때 최단 시간에 다른 정점에 도달 할 수 있습니다.



* 위 문제는 백준의 14940 ‘쉬운 최단 거리’라는 문제입니다. 위 문제는 BFS로 풀 수 있는데 격자 그래프에서 BFS를 사용했을 때의 방문 순서를 잘 보여줍니다. 2에서 출발 해서 0은 갈 수 없는 길이고 1은 갈 수 있는 길을 나타내고 있습니다. 출발 지점에서 점차 멀어지며 출발 지점에서 해당 거리까지의 가장 최단 거리를 잘 나타내고 있습니다.
* 이게 가능한 이유는 큐라는 자료구조에 현재 정점을 기준으로 방문할 정점 정보를 저장하고, 다음에 이 큐에서 하나를 꺼내서 정점을 탐색하고 꺼낸 정점을 기주으로 다음에 방문할 정점을 큐에 넣기 때문입니다. 큐이기 때문에 정점에서 가까운 것들은 꺼내기 전까지 계속 쌓여 있는 것이죠.
* 위 그림은 BFS를 그림으로 나타낸 한 예시입니다. 각 정점마다 적힌 숫자는 방문 순서를 적은 것인데, 빨간 색으로 붂인 부분은 프로그래밍을 어떻게 하느냐에 따라 순서가 달라질 수 있습니다. 출발 정점에서 점차 멀어지는 것을 확인하실 수 있으며, 각 간선에 가중치가 없을 경우 빨간색으로 묶인 부분을 한 번씩 거치면 해당 노드로 가는 최단 거리임을 확인할 수 있습니다.

공통 문제

그래프를 DFS로 탐색한 결과와 BFS로 탐색한 결과를 출력하는 프로그램을 작성하시오. 단, 방문할 수 있는 정점이 여러 개인 경우에는 정점 번호가 작은 것을 먼저 방문하고, 더 이상 방문할 수 있는 점이 없는 경우 종료한다. 정점 번호는 1번부터 N번까지이다.

첫째 줄에 정점의 개수 N(1 ≤ N ≤ 1,000), 간선의 개수 M(1 ≤ M ≤ 10,000), 탐색을 시작할 정점의 번호 V가 주어진다. 다음 M개의 줄에는 간선이 연결하는 두 정점의 번호가 주어진다. 어떤 두 정점 사이에 여러 개의 간선이 있을 수 있다. 입력으로 주어지는 간선은 양방향이다.

첫째 줄에 DFS를 수행한 결과를, 그 다음 줄에는 BFS를 수행한 결과를 출력한다. V부터 방문된 점을 순서대로 출력하면 된다.

해결 과정