08. Graph 문제풀이

Div. 3 알고리즘 스터디 / 임지환

Problem set

- 1260 DFS와 BFS
- 1012 유기농 배추
- 2583 영역 구하기
- 11724 연결 요소의 개수
- 13023 ABCDE
- 10271 Beam me out!*
- 2178 미로 탐색
- 7562 나이트의 이동
- 2644 촌수계산
- 5427 불*
- 2206 벽 부수고 이동하기*
- 9019 DSLR*

#1260 DFS와 BFS

- DFS 짜보고 BFS 짜보라는 문제.
- "정점 번호가 작은 것을 먼저 방문"

http://boj.kr/6da17ad7abdb4e7593663afd05be4ef8

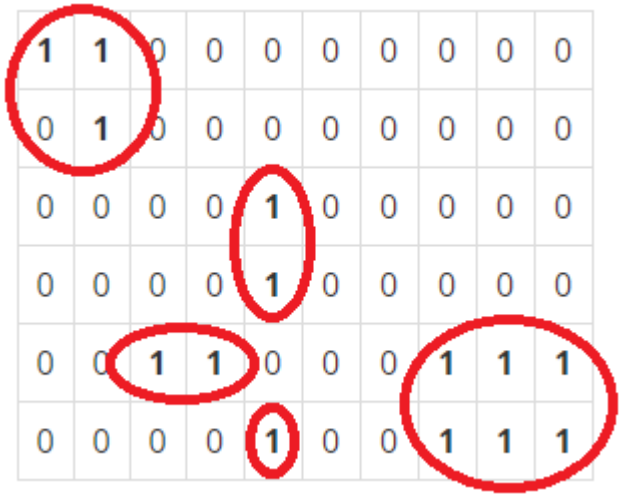
#1012 유기농 배추

- 정점에 해당할 것들이 (x, y)형태의 좌표로 표현될 수 있으므로 굳이 vector을 통한 인접리스트를 만들 필요 없다.
- int arr[50][50];
- 인접한 노드로의 이동 : 현재 좌표에서 (-1, 0), (0, -1), (1, 0), (0, 1)만큼 이동

#1012 유기농 배추

- 1로 이루어진 덩어리의 개수를 세는 문제

  ⇒모든 좌표에 대해 방문하지 않은 1을 만났을 때 dfs 혹은 bfs



http://boj.kr/97d6bc9634e84becb4e8da141269e01b

#2583 영역 구하기

- 주어진 직사각형들에 대한 표시를 하고, 표시가 되지 않은 영역을 count
- 순회 방식은 유기농 배추와 동일.

#11724 연결요소의 개수

- 입력 예시 :

  6 5

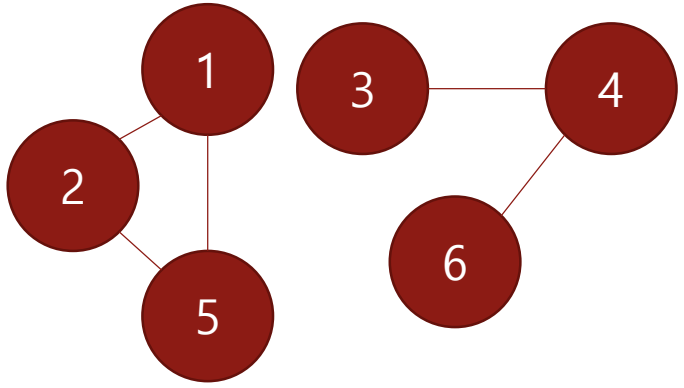
  1 2

  2 5

  5 1

  3 4

  4 6



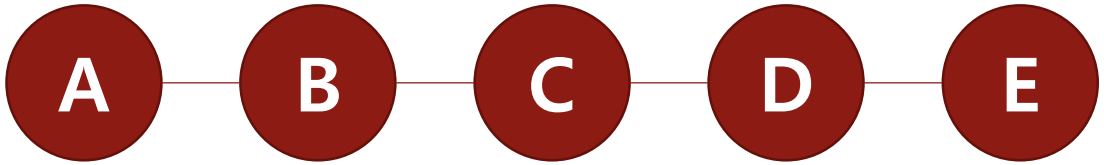
#11724 연결요소의 개수

- 한데 묶인 덩어리의 개수를 구하는 문제
- 그래프 순회가 총 몇번 일어나는지를 count

http://boj.kr/3a3ebc5a114e400289387c9435204f49

#13023 ABCDE

- 아래와 같은 그래프 연결관계가 있는지를 확인

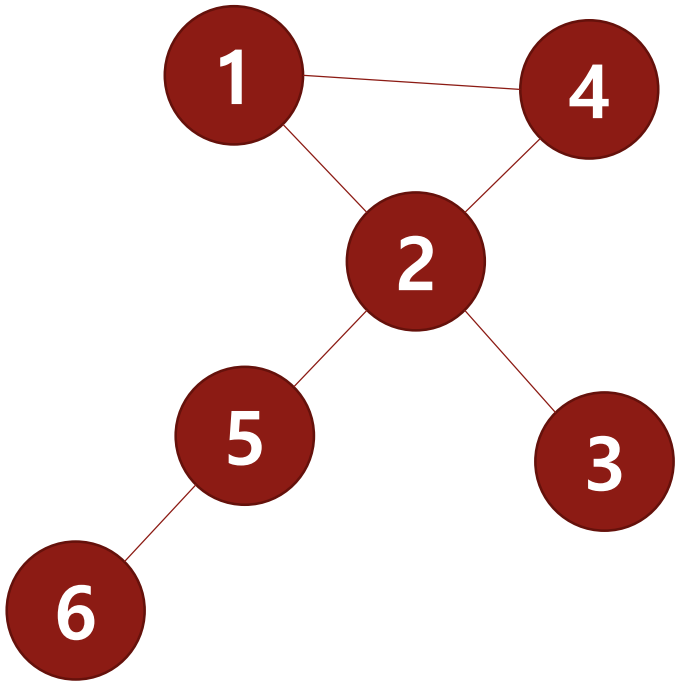


⇒특정 노드부터 탐색을 시작하여 깊이 5만큼 내려갈 수 있는가?

#13023 ABCDE

- 입력 예시 :

```
6 6
1 2
2 3
1 4
2 4
2 5
5 6
```

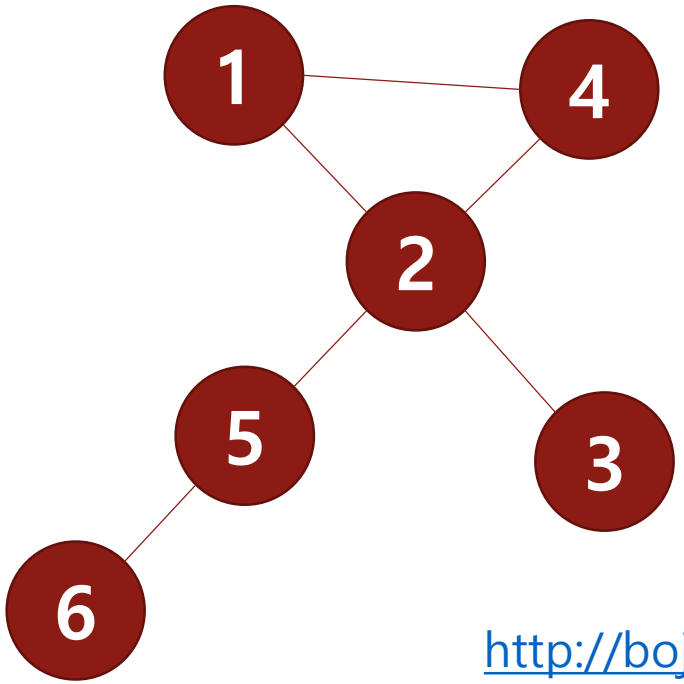


- 1 -> 2 -> 3 은 답이 될 수 없다.
- 그렇다고 2를 방문처리 해버리면 1->4->2->5->6으로 답을 만들 수 없다.

#13023 ABCDE

- 입력 예시 :

```
6 6
1 2
2 3
1 4
2 4
2 5
5 6
```



- 1 -> 2 -> 3 은 답이 될 수 없다.
- 그렇다고 2를 방문처리 해버리면 1->4->2->5->6으로 답을 만들 수 없다.
- 1->2->3 에서 경로 탐색이 끝나면 방문처리를 해제해준다.

http://boj.kr/623608af3a4b461ea39e86b435b2ba78

#10271 Beam me out!

- 안 풀고 풀이 보시는거 다 압니다. 일단 문제부터 읽고 오시지요^^

풀이 : http://raararaara.blog.me/221522001287

#2178 미로 탐색

- 도착지점까지 가기 위해 지나야 하는 **최소의 칸 수** 출력

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

=

1	0	9	10	11	12
2	0	8	0	12	0
3	0	7	0	13	14
4	5	6	0	14	15

#2178 미로 탐색

- 방문을 완료한 지점에 몇번만에 이동했는지에 대한 값을 저장
- BFS를 통해 방문한 지점을 또 방문하게 되는 경우 기존에 저장된 값보다 더 작은 값으로 방문을 할 수 있을까?

http://boj.kr/624064accb6f46ba9c25a2e9c2fb324e

#7562 나이트의 이동

- 2178 미로탐색과 진행 방식이 유사.
- 시작 지점으로부터

  1번의 이동으로 갈 수 있는 위치

  2번의 이동으로 갈 수 있는 위치

  …

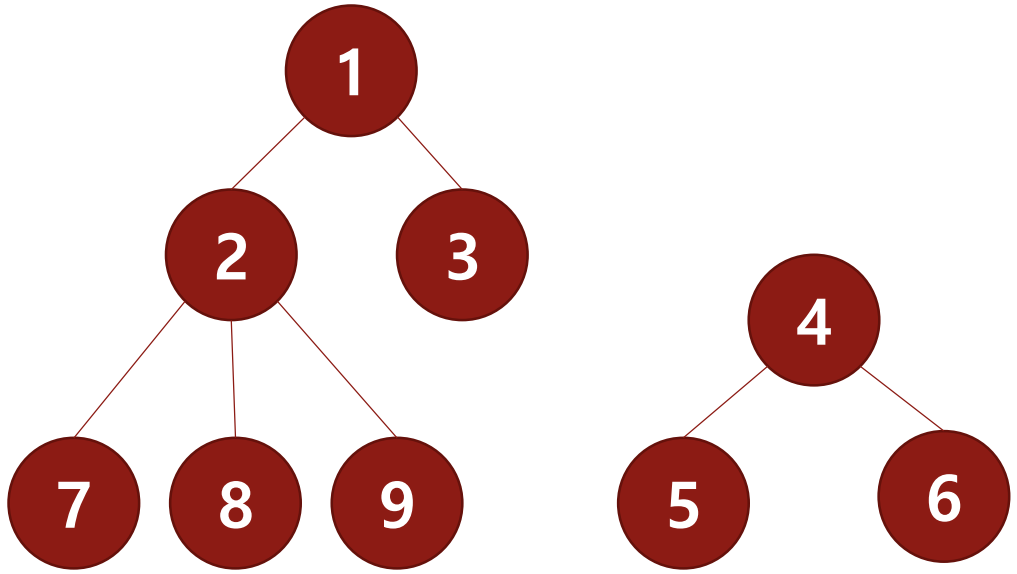
  k번의 이동 끝에 목적지에 도달한다면 탐색 종료

http://boj.kr/c23108dd4fd545c1a009e26825e54bed

#2644 촌수계산

- 입력 예시 :

```
9
7 3
7
1 2
1 3
2 7
2 8
2 9
4 5
4 6
```



#2644 촌수계산

- 부모는 반드시 한명일 수밖에 없다.

  ⇒트리

- 아래에서 위로 가는 경로는 하나일 수밖에 없고,
- 마찬가지로 위에서 아래로 가는 경로 또한 하나일 수밖에 없다.

http://boj.kr/319120c1e0e540039a5db6009732a6f9

#2206 벽 부수고 이동하기

int vis[1000][1000][2]

- 벽을 한번만 부술 수 있기 때문에, 특정 좌표를 방문할 때
  - 한번이라도 벽을 부수고 방문한건지(vis[r][c][1])
  - 벽을 부순적 없이 방문한건지에 대한 확인이 필요하다.(vis[r][c][0])
- 앞의 미로탐색과 마찬가지로 방문여부를 체크하는게 아닌 몇번의 이동으로 방문했는지를 저장

#2206 벽 부수고 이동하기

- 벽을 부순적이 있는데, 다음 방문할 곳이 벽이라면 이동 불가
- 최종 목적지에 대해

  벽을 부순적 없이 방문하는 경우와

  벽을 부순 후 방문하는 경우 중 이동 경로가 작은 값을 답으로 한다.

http://boj.kr/21875a4629524fc09bc0faf0b1e11b6b

#5427 불

- 불의 확장과 사람의 이동은 동시에 일어나긴 하지만, 불이 확장될 곳으로는 이동할 수 없다.
- 불의 확장과 사람의 이동 모두 한 cycle에 처리하되, 불의 확장을 먼저 처리해주면 위의 경우를 방지 가능.

http://boj.kr/2f07137e21ba4cffb9c1abf84d29b94c

#9019 DSLR

- 문제 풀어보고 보는거 맞죠?
- 레지스터에 저장될 수 있는 수는 0이상 10000미만이므로 가능한 수의 종류가 10000가지이다. 즉 10000개의 노드가 존재할 수 있다.
- "최소한"의 명령어 나열

#9019 DSLR

- 암시적 그래프 문제.
- A에서 B가 되는 수들의 경로를 찾는 문제로, 방문 여부를 체크하는 방식에서 더 나아가 현재 수가 되기 이전의 수를 저장해준다.

  ex) 1234 -> 2341 -> 3412에서

  prev[2341] = 1234, prev[3412] = 2341
- 이전 수 말고 다음 수를 저장하는 방식으로 풀어도 되지만, 이전 수를 저장했을 경우 출력이 역순으로 되어야 하기 때문에 스택을 사용.

http://boj.kr/d7bf9a4bc7624bb59bda87ef83df1312