## 1. 가장 먼 노드

### 문제 설명

n 개의 노드가 있는 그래프가 있습니다. 각 노드는 1 부터 n 까지 번호가 적혀있습니다. 1 번 노드에서 가장 멀리 떨어진 노드의 갯수를 구하려고 합니다. 가장 멀리 떨어진 노드란 최단경로로 이동했을 때 간선의 개수가 가장 많은 노드들을 의미합니다.

노드의 개수 n, 간선에 대한 정보가 담긴 2 차원 배열 vertex 가 매개변수로 주어질 때, 1 번 노드로부터 가장 멀리 떨어진 노드가 몇 개인지를 return 하도록 solution 함수를 작성해주세요.

제한사항

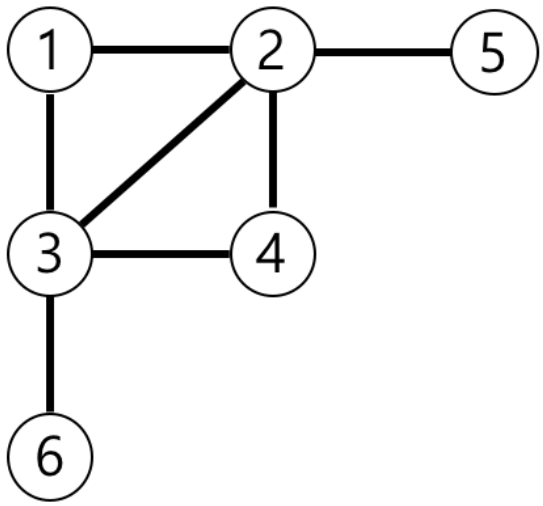
- 노드의 개수 n 은 2 이상 20,000 이하입니다.
- 간선은 양방향이며 총 1 개 이상 50,000 개 이하의 간선이 있습니다.
- vertex 배열 각 행 [a, b]는 a 번 노드와 b 번 노드 사이에 간선이 있다는 의미입니다.

입출력 예

| n | vertex | return |
|---|---|---|
| 6 | [[3, 6], [4, 3], [3, 2], [1, 3], [1, 2], [2, 4], [5, 2]] | 3 |

입출력 예 설명

예제의 그래프를 표현하면 아래 그림과 같고, 1 번 노드에서 가장 멀리 떨어진 노드는 4,5,6 번 노드입니다.



```
import java.util.*;
class Solution {
    public int solution(int n, int[][] edge) {


        int dist[] = new int[n+1];
        boolean visit[][] = new boolean[n+1][n+1];
        for(int i=0; i<edge.length; i++){
            visit[edge[i][0]][edge[i][1]] = true;
            visit[edge[i][1]][edge[i][0]] = true;
        }

        Queue<Integer> queue = new LinkedList();
        queue.add(1);
        int max = 0;

        while(!queue.isEmpty()){
            int idx = queue.poll();
            for(int j=2; j<=n; j++){
                if(dist[j] == 0 && visit[idx][j]){
                    dist[j] = dist[idx] + 1;
                    queue.add(j);
                }
            }
        }
        for(int i=0; i<n+1; i++){
            max = Math.max(max, dist[i]);
        }
        int cnt = 0;
        for(int i=0; i<n+1; i++){
            if(max == dist[i])
                cnt++;
        }

        return cnt;
    }
}
```

https://ukyonge.tistory.com/199

## 2. 순위

문제 설명

n 명의 권투선수가 권투 대회에 참여했고 각각 1 번부터 n 번까지 번호를 받았습니다. 권투 경기는 1 대 1 방식으로 진행이 되고, 만약 A 선수가 B 선수보다 실력이 좋다면 A 선수는 B 선수를 항상 이깁니다. 심판은 주어진 경기 결과를 가지고 선수들의 순위를 매기려 합니다. 하지만 몇몇 경기 결과를 분실하여 정확하게 순위를 매길 수 없습니다.

선수의 수 n, 경기 결과를 담은 2 차원 배열 results 가 매개변수로 주어질 때 정확하게 순위를 매길 수 있는 선수의 수를 return 하도록 solution 함수를 작성해주세요.

제한사항

- 선수의 수는 1 명 이상 100 명 이하입니다.
- 경기 결과는 1 개 이상 4,500 개 이하입니다.
- results 배열 각 행 [A, B]는 A 선수가 B 선수를 이겼다는 의미입니다.
- 모든 경기 결과에는 모순이 없습니다.

입출력 예

| n | results | return |
|---|---|---|
| 5 | [[4, 3], [4, 2], [3, 2], [1, 2], [2, 5]] | 2 |

입출력 예 설명

2 번 선수는 [1, 3, 4] 선수에게 패배했고 5 번 선수에게 승리했기 때문에 4 위입니다
5 번 선수는 4 위인 2 번 선수에게 패배했기 때문에 5 위입니다.

```
public int solution(int n, int[][] results) {
        int answer = 0;
        int maxValue= 1000000;
        int[][] FW = new int[n+1][n+1];
        for(int i = 1; i <=n; i++) {
            for(int j = 1; j<=n; j++) {
                FW[i][j] = maxValue;
            }
        }

        for(int[] e : results) {
            FW[e[0]][e[1]] = 1;
        }
        //거치는 노드
        for(int k = 1; k <= n; k++) {
            for(int i = 1; i <=n; i++) {            //시작 노드
```

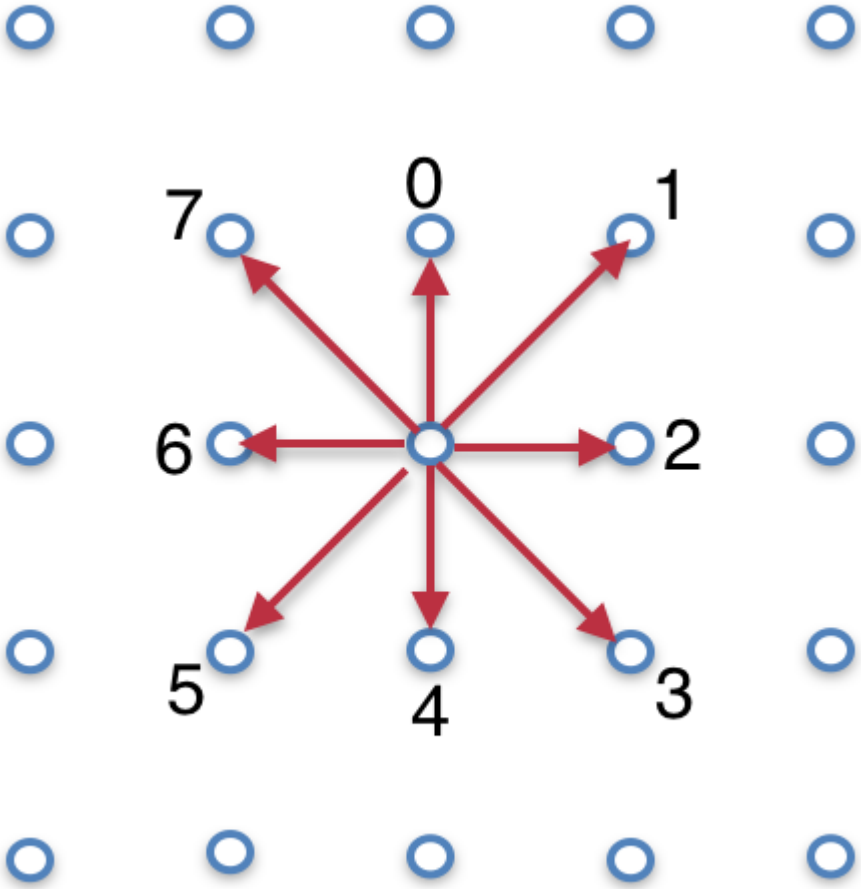
```
                for(int j = 1; j <= n; j++) {
                    if(FW[i][j] > FW[i][k]+FW[k][j]) {    //종료(
도착)노드
                        FW[i][j] = FW[i][k] + FW[k][j];
                    }
                }
            }
        }

        for(int i = 1; i <= n; i++){
            boolean flag = true;
            for(int j = 1; j <=n; j++) {
                if(i == j) continue;
                if(FW[i][j] == maxValue && FW[j][i] == maxValue)
 {
                    flag = false;
                    break;
                }
            }
            if(flag) answer++;
        }
        return answer;
    }
```

https://tosuccess.tistory.com/47

## 3. 방의 개수

### 문제 설명

원점(0,0)에서 시작해서 아래처럼 숫자가 적힌 방향으로 이동하며 선을 긋습니다.




ex) 1 일때는 오른쪽 위로 이동

그림을 그릴 때, 사방이 막히면 방하나로 샙니다.
이동하는 방향이 담긴 배열 arrows 가 매개변수로 주어질 때, 방의 갯수를 return 하도록 solution 함수를 작성하세요.

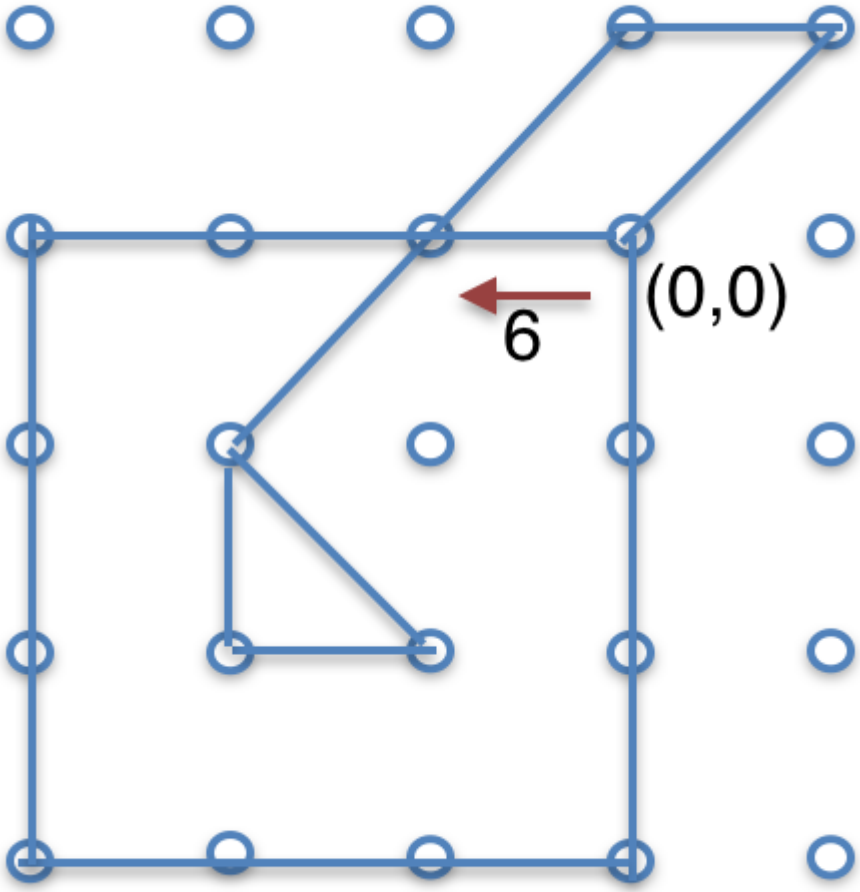
제한사항

- 배열 arrows 의 크기는 1 이상 100,000 이하 입니다.
- arrows 의 원소는 0 이상 7 이하 입니다.
- 방은 다른 방으로 둘러 싸여질 수 있습니다.

입출력 예

| arrows | return |
| --- | --- |
| [6, 6, 6, 4, 4, 4, 2, 2, 2, 0, 0, 0, 1, 6, 5, 5, 3, 6, 0] | 3 |

입출력 예 설명




- (0,0) 부터 시작해서 6(왼쪽) 으로 3 번 이동합니다. 그 이후 주어진 arrows 를 따라 그립니다.
- 삼각형 (1), 큰 사각형(1), 평행사변형(1) = 3

출처

```
from collections import deque

dx = [-1, -1, 0, 1, 1, 1, 0, -1]
dy = [0, 1, 1, 1, 0, -1, -1, -1]

def solution(arrows):
    cnt, dir, q = {}, {}, deque()
    cnt[(0, 0)] = 0
    q.append([0, 0])
    x, y, ans = 0, 0, 0

    for i in arrows:
        for j in range(2):
            nx = x + dx[i]
            ny = y + dy[i]
            cnt[(nx, ny)] = 0
            dir[(x, y, nx, ny)] = 0
            dir[(nx, ny, x, y)] = 0
            q.append([nx, ny])
            x, y = nx, ny

    x, y = q.popleft()
    cnt[(x, y)] = 1
    while q:
        nx, ny = q.popleft()

        if cnt[(nx, ny)] == 1:
            if dir[(x, y, nx, ny)] == 0:
                ans += 1
                dir[(x, y, nx, ny)] = 1
                dir[(nx, ny, x, y)] = 1
        else:
            cnt[(nx, ny)] = 1
            dir[(x, y, nx, ny)] = 1
            dir[(nx, ny, x, y)] = 1

        x, y = nx, ny

    return ans
```

https://chldkato.tistory.com/101

자바 없음 ..