7576 - 토마토

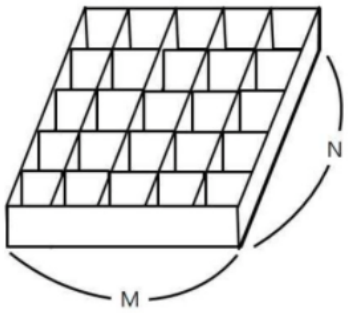
난이도	실버 1

문제

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	256 MB	103779	37063	23226	34.136%

문제

철수의 토마토 농장에서는 토마토를 보관하는 큰 창고를 가지고 있다. 토마토는 아래의 그림과 같이 격자 모양 상자의 칸에 하나씩 넣어서 창고에 보관한다.



창고에 보관되는 토마토들 중에는 잘 익은 것도 있지만, 아직 익지 않은 토마토들도 있을 수 있다. 보관 후 하루가 지나면, 익은 토마토들의 인접한 곳에 있는 익지 않은 토마토들은 익은 토마토의 영향을 받아 익게 된다. 하나의 토마토의 인접한 곳은 왼쪽, 오른쪽, 앞, 뒤 네 방향에 있는 토마토를 의미한다. 대각선 방향에 있는 토마토들에게는 영향을 주지 못하며, 토마토가 혼자 저절로 익는 경우는 없다고 가정한다. 철수는 창고에 보관된 토마토들이 며칠이 지나면 다 익게 되는지, 그 최소 일수를 알고 싶어 한다.

토마토를 창고에 보관하는 격자모양의 상자들의 크기와 익은 토마토들과 익지 않은 토마토들의 정보가 주어졌을 때, 며칠이 지나면 토마토들이 모두 익는지, 그 최소 일수를 구하는 프로그램을 작성하라. 단, 상자의 일부 칸에는 토마토가 들어있지 않을 수도 있다.

입력

첫 줄에는 상자의 크기를 나타내는 두 정수 M,N이 주어진다. M은 상자의 가로 칸의 수, N은 상자의 세로 칸의 수를 나타낸다. 단, 2 ≤ M,N ≤ 1,000 이다. 둘째 줄부터는 하나의 상자에 저장된 토마토들의 정보가 주어진다. 즉, 둘째 줄부터 N개의 줄에는 상자에 담긴 토마토의 정보가 주어진다. 하나의 줄에는 상자 가로줄에 들어있는 토마토의 상태가 M개의 정수로 주어진다. 정수 1은 익은 토마토, 정수 0은 익지 않은 토마토, 정수 -1은 토마토가 들어있지 않은 칸을 나타낸다.

토마토가 하나 이상 있는 경우만 입력으로 주어진다.

출력

여러분은 토마토가 모두 익을 때까지의 최소 날짜를 출력해야 한다. 만약, 저장될 때부터 모든 토마토가 익어있는 상태이면 0을 출력해야 하고, 토마토가 모두 익지는 못하는 상황이면 -1을 출력해야 한다.

풀이과정

최소 날짜라는 키워드를 보고 **BFS**를 떠올렸다. 기본적인 BFS 문제와 다른 점이 하나 있다면, 시작점이 여러 개였다는 것이다. 이 문제를 풀면서, 시작점이 여러 개면서 최소 날짜 혹은 최단 거리 등을 뽑아내려면 어떻게 접근해야 하는지 배웠다.

그리고 이전에는 무조건 visited 배열을 만들어 방문 여부를 체크해주는 배열을 따로 만들어야겠다는 생각이었는데, 이 문제에서는 굳이 저 배열을 만들지 않아도 체크해줄 수 있어서 이전의 생각이 좀 바뀌게 되는 문제이기도 했다.

소스코드

```
package BOJ7576;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.LinkedList;
import java.util.Queue;
import java.util.StringTokenizer;

class Tomato {
    int x;
    int y;

    Tomato(int x, int y){
        this.x = x;
        this.y = y;
    }
}

public class Main {

    static int N, M; // 2 <= M,N <= 1000
    static int[][] tomato;
    static Queue<Tomato> q;
//  static boolean[][] visited;
    static int[] dx = {-1, 1, 0, 0};
    static int[] dy = {0, 0, -1, 1};

    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        StringTokenizer st = new StringTokenizer(br.readLine());
        M = Integer.parseInt(st.nextToken()); // 가로
        N = Integer.parseInt(st.nextToken()); // 세로

        tomato = new int[N][M];
        q = new LinkedList<>();

        for(int i=0 ; i<N ; i++){
            st = new StringTokenizer(br.readLine());
            for(int j=0 ; j<M ; j++)
```

```
                tomato[i][j] = Integer.parseInt(st.nextToken());
        }

        // 익은 토마토는 Queue에 넣어주자
        for(int i=0 ; i<N ; i++){
            for(int j=0 ; j<M ; j++){
                if(tomato[i][j] == 1) {
                    q.add(new Tomato(i, j));
                }
            }
        }

        System.out.println(bfs());
    }

    static int bfs() {
        int nx, ny;
        while(!q.isEmpty()) {
            Tomato cur = q.poll();
            for(int i=0 ; i<4 ; i++){
                nx = cur.x + dx[i];
                ny = cur.y + dy[i];

                // 배열을 벗어나거나 이미 방문했거나, 혹은 방문할 수 없는 경우 continue
                // 이미 익은 경우도 continue
                if(nx < 0 || ny < 0 || nx >= N || ny >= M) continue;
                if(tomato[nx][ny] != 0) continue;

                tomato[nx][ny] = tomato[cur.x][cur.y] + 1;
                q.add(new Tomato(nx, ny));
            }
        }

        int answer = Integer.MIN_VALUE;
        for(int i=0 ; i<N ; i++){
            for(int j=0 ; j<M ; j++) {
                if(tomato[i][j] == 0) return -1;

                answer = Math.max(answer, tomato[i][j]);
            }
        }

        if(answer == 1) return 0;
        else return answer - 1;

    }
}
```

💡 주요 포인트

1. 익은 토마토는 하나가 아니다.

그렇다면, 익은 토마토에서 동시에 출발할 수 있도록 익은 토마토들을 큐에 넣어주자.

2. 방문 여부를 체크해주는 배열이 없어도 익지 않은 토마토들을 체크해줄 수 있다.

토마토가 없는 칸은 -1, 익은 토마토는 1로 표현되고, 익지 않은 토마토들만 0으로 표현된다.

즉 처음엔 익지 않은 토마토였어도, 방문을 했다면 익은 토마토가 되어 1로 바뀌었을 것이다. 또한 모든 방문이 끝나고나서 방문하지 않은 곳이 있는지, 즉 익지 않은 토마토가 있는지 검사하려면 해당 토마토 배열에 0이 남아있는지 검사해주면 된다.

> 사담

처음에는 큐의 제네릭을 int 배열로 정해서 토마토의 좌표를 저장해주려고 했는데,

```
tomato[nx][ny] = tomato[cur[0]][cur[1]] + 1;
```

저 부분 코드가 더러워보여서 그냥 따로 Tomato 클래스를 만들어주었다.