Graph Coloring

|  |
| --- |
| Program: color.(py|cpp|java)  Input: color.in  Output: color.out |

**Description**

A graph coloring problem involves assigning a color to each node such that no two adjacent nodes assigned the same color. See the below example for a colored graph.

A picture containing blur

Description automatically generated

Graph coloring has a wide range of applications including register allocation in compiler design, scheduling, and even a solution for the SODUKO puzzle1.  
  
Your task is to design a backtracking algorithm that finds a *minimum* color assignment for all nodes in an undirected graph .

Input

The first line on the input file contains an integer denoting the number of test cases. In each test case, you are given an adjacency matrix representing graph .   
Each test case starts with two positive integers on a separate line followed by adjacency matrix representing graph .

Output

For each test case, print on a separate line the minimum number of colors required to solve graph coloring problem.

Additional Deliverables

1. Time and space analysis for the algorithm part only.
2. Is there any kind of input that we can easily solve?
3. What is the maximum depth of the backtracking tree for any given input graph?

1 https://en.wikipedia.org/wiki/Graph\_coloring#Applications

**Sample Input /output**

color.out

3

3 3

0 1 1

1 0 0

1 0 0

4 4

0 1 1 1

1 0 1 1

1 1 0 1

1 1 1 0

2 2

1 0

0 1

2

4

2

color.in