Started on	Wednesday, 30 April 2025, 11:18 AM
State	Finished
Completed on	Wednesday, 30 April 2025, 12:09 PM
Time taken	51 mins 4 secs
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 10.00 out of 10.00

GRAPH COLORING PROBLEM

Given an undirected graph and a number m, determine if the graph can be coloured with at most m colours such that no two adjacent vertices of the graph are colored with the same color. Here coloring of a graph means the assignment of colors to all vertices.

Input-Output format:

Input:

1. A 2D array graph[V][V] where V is the number of vertices in graph and graph[V][V] is an adjacency matrix representation of the graph. A value graph[i][j] is 1 if there is a direct edge from i to j, otherwise graph[i][j] is 0.
2. An integer m is the maximum number of colors that can be used.

Output:

An array color[V] that should have numbers from 1 to m. color[i] should represent the color assigned to the ith vertex.

Example:

```
Input:
graph = {0, 1, 1, 1},
        {1, 0, 1, 0},
        {1, 1, 0, 1},
        {1, 0, 1, 0}
Output:
Solution Exists:
Following are the assigned colors
 1  2  3  2
Explanation: By coloring the vertices
with following colors, adjacent
vertices does not have same colors
```

```
Input:
graph = {1, 1, 1, 1},
        {1, 1, 1, 1},
        {1, 1, 1, 1},
        {1, 1, 1, 1}
Output: Solution does not exist.
Explanation: No solution exits.
```

Answer: (penalty regime: 0 %)

```
class Graph:
    def __init__(self,vertices):
        self.v=vertices
        self.graph=[[0 for column in range(vertices)] for row in range(vertices)]
    def isSafe(self,v,colour,c):
        for i in range(self.v):
            if self.graph[v][i]==1 and colour[i]==c:
                return False
        return True
    def graphColouringUtil(self,m,colour,v):
        if v==self.v:
            return True
        for c in range(1,m+1):
            if self.isSafe(v,colour,c):
                colour[v]=c
                if self.graphColouringUtil(m,colour,v+1):
                    return True
                colour[v]=0
        return False
    def graphColouring(self,m):
        colour=[0]*self.v
```

	Test	Expected	Got	

✓	`g = Graph(4)`
	`g.graph = [[0, 1, 1, 1], [1, 0, 1, 0], [1, 1, 0, 1], [1, 0, 1, 0]]`
	`m = 3`
	`g.graphColouring(m)`
		`Solution exist and Following are the assigned colours:`
		`1 2 3 2`
			`Solution exist and Following are the assigned colours:`
			`1 2 3 2`
				✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.