```
def is_safe(graph, color, v, c, n):
    for i in range(n):
        if graph[v][i] == 1 and color[i] == c:
            return False
    return True
def graph_coloring(graph, color, v, m, n):
    if v == n:
        return True
    for c in range(1, m + 1):
        if is_safe(graph, color, v, c, n):
            color[v] = c
            if graph_coloring(graph, color, v + 1, m, n):
                return True
            color[v] = 0
    return False
def solve_graph_coloring(graph, m, n):
    color = [0] * n
    if not graph_coloring(graph, color, 0, m, n):
        print("Solution does not exist")
    else:
        print("Solution found:")
        color_names = {1: "red", 2: "blue", 3: "green", 4: "yellow", 5: "purple", 6: "orange", 7:
"pink"}
        for v in range(n):
            color_name = color_names.get(color[v], "unknown")
            print(f"Vertex {v} ---> {color_name}")
def main():
    n = int(input("Enter the number of vertices: "))
    print("Enter the adjacency matrix (one row at a time, space-separated):")
    graph = []
    for _ in range(n):
        row = list(map(int, input().split()))
        graph.append(row)
    m = int(input("Enter the number of colors: "))
    solve_graph_coloring(graph, m, n)


if __name__ == "__main__":
    main()
```

Output:

```
Enter the number of vertices: 4
Enter the adjacency matrix (one row at a time, space-separated):
0 1 1 0
1 0 1 0
1 1 0 1
1 0 1 0
Enter the number of colors: 3
Solution found:
Vertex 0 ---> red
Vertex 1 ---> blue
Vertex 2 ---> green
Vertex 3 ---> blue
```