TASK.6

def is\_safe(v, graph, color, c):

"""

Checks if assigning color 'c' to vertex 'v' is safe.

It's safe if no adjacent vertex has the same color.

"""

for i in range(len(graph)):

if graph[v][i] == 1 and color[i] == c:

return False

return True

def graph\_coloring\_util(graph, m, color, v):

"""

Recursive utility function to solve the graph coloring problem.

'graph': Adjacency matrix of the graph.

'm': Maximum number of colors allowed.

'color': List to store colors assigned to vertices.

'v': Current vertex.

"""

num\_vertices = len(graph)

# Base case: all vertices are assigned a color

if v == num\_vertices:

print("Solution exists with colors:", color)

return True

# Try assigning each color to this vertex

for c in range(1, m + 1):

if is\_safe(v, graph, color, c):

color[v] = c

if graph\_coloring\_util(graph, m, color, v + 1):

return True

color[v] = 0 # Backtrack

return False

def solve\_graph\_coloring(graph, m):

"""

Solves the graph coloring problem.

'graph': Adjacency matrix of the graph.

'm': Maximum number of colors allowed.

"""

num\_vertices = len(graph)

color = [0] \* num\_vertices # Initialize all vertices as uncolored

if not graph\_coloring\_util(graph, m, color, 0):

print("Solution does not exist with", m, "colors.")

# Example Usage:

if \_\_name\_\_ == "\_\_main\_\_":

# Example graph 1

graph1 = [

[0, 1, 1, 1],

[1, 0, 0, 1],

[1, 0, 0, 1],

[1, 1, 1, 0]

]

max\_colors1 = 3

print(f"Graph 1 with {max\_colors1} colors:")

solve\_graph\_coloring(graph1, max\_colors1)

print("\n" + "="\*30 + "\n")

# Example graph 2

graph2 = [

[0, 1, 0, 1],

[1, 0, 1, 0],

[0, 1, 0, 1],

[1, 0, 1, 0]

]

max\_colors2 = 2

print(f"Graph 2 with {max\_colors2} colors:")

solve\_graph\_coloring(graph2, max\_colors2)

print("\n" + "="\*30 + "\n")

# Example graph 3 (no valid solution with given colors)

graph3 = [

[0, 1, 1],

[1, 0, 1],

[1, 1, 0]

]

max\_colors3 = 2

print(f"Graph 3 with {max\_colors3} colors:")

solve\_graph\_coloring(graph3, max\_colors3)

OUTPUT:

A screenshot of a computer

AI-generated content may be incorrect.