**Roll No: 3263**

**Assignment:4**

**Title:** Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and

Backtracking for n-queens problem or a graph coloring problem.

class Graph:

def \_\_init\_\_(self, vertices):

self.vertices = vertices

self.adj\_matrix = [[0 for \_ in range(vertices)] for \_ in range(vertices)]

def add\_edge(self, u, v):

self.adj\_matrix[u][v] = 1

self.adj\_matrix[v][u] = 1

def is\_safe(self, v, color, colors):

for i in range(self.vertices):

if self.adj\_matrix[v][i] == 1 and colors[i] == color:

return False

return True

def graph\_coloring\_util(self, num\_colors, colors, v):

if v == self.vertices:

return True

for color in range(1, num\_colors + 1):

if self.is\_safe(v, color, colors):

colors[v] = color

if self.graph\_coloring\_util(num\_colors, colors, v + 1):

return True

colors[v] = 0

return False

def graph\_coloring(self, num\_colors):

colors = [0] \* self.vertices

if not self.graph\_coloring\_util(num\_colors, colors, 0):

print("No solution exists.")

else:

print("Graph coloring solution:")

for v in range(self.vertices):

print(f"Vertex {v + 1} -> Color {colors[v]}")

# Get the graph details from the user

num\_vertices = int(input("Enter the number of vertices: "))

num\_edges = int(input("Enter the number of edges: "))

graph = Graph(num\_vertices)

print("Enter the edges in the format 'u v', where u and v are vertices connected by an edge:")

for \_ in range(num\_edges):

u, v = map(int, input().split())

# Validate the vertices entered

if u < 1 or u > num\_vertices or v < 1 or v > num\_vertices:

print("Invalid vertex entered. Please try again.")

exit(1)

graph.add\_edge(u - 1, v - 1)

num\_colors = int(input("Enter the number of colors available: "))

# Solve the graph coloring problem

graph.graph\_coloring(num\_colors)

output:

PS D:\AI assignments> d:; cd 'd:\AI assignments'; & 'C:\Program Files\Python311\python.exe' 'c:\Users\Dell\.vscode\extensions\ms-python.python-2023.6.1\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '59284' '--' 'd:\AI assignments\csp.py'

Enter the number of vertices: 5

Enter the number of edges: 6

Enter the edges in the format 'u v', where u and v are vertices connected by an edge:

1 2

1 4

1 3

4 5

2 3

3 5

Enter the number of colors available: 3

Graph coloring solution:

Vertex 1 -> Color 1

Vertex 2 -> Color 2

Vertex 3 -> Color 3

Vertex 4 -> Color 2

Vertex 5 -> Color 1

PS D:\AI assignments>