Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for a graph coloring problem.

Code:

#include <bits/stdc++.h>

using namespace std;

#define V 4

void printSolution(int color[]);

bool isSafe(bool graph[V][V], int color[])

{

for (int i = 0; i < V; i++)

for (int j = i + 1; j < V; j++)

if (graph[i][j] && color[j] == color[i])

return false;

return true;

}

bool graphColoring(bool graph[V][V], int m, int i,

int color[V])

{

if (i == V) {

if (isSafe(graph, color)) {

printSolution(color);

return true;

}

return false;

}

for (int j = 1; j <= m; j++) {

color[i] = j;

if (graphColoring(graph, m, i + 1, color))

return true;

color[i] = 0;

}

return false;

}

void printSolution(int color[])

{

cout << "Solution Exists:"

" Following are the assigned colors \n";

for (int i = 0; i < V; i++)

cout << " " << color[i];

cout << "\n";

}

int main()

{

bool graph[V][V] = {

{ 0, 1, 1, 1 },

{ 1, 0, 1, 0 },

{ 1, 1, 0, 1 },

{ 1, 0, 1, 0 },

};

int m = 3;

int color[V];

for (int i = 0; i < V; i++)

color[i] = 0;

if (!graphColoring(graph, m, 0, color))

cout << "Solution does not exist";

return 0;

}

Output:

Solution Exists: Following are the assigned colors

1 2 3 2