1. **Graph colouring using backtracking**

**Code:**

#include <stdio.h>

#include <stdbool.h>

#define MAX\_VERTICES 10

// Function to print the solution

void printSolution(int color[], int V) {

printf("Vertex\tColor\n");

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

printf("%d\t%d\n", i, color[i]);

}

}

// Function to check if the current color assignment is safe for the vertex v

bool isSafe(int v, bool graph[][MAX\_VERTICES], int color[], int c, int V) {

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

if (graph[v][i] && c == color[i]) {

return false;

}

}

return true;

}

// Recursive function to solve the graph coloring problem using backtracking

bool graphColoringUtil(bool graph[][MAX\_VERTICES], int m, int color[], int v, int V) {

if (v == V) {

return true; // All vertices are assigned a color

}

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

if (isSafe(v, graph, color, c, V)) {

color[v] = c;

// Recur to assign colors to rest of the vertices

if (graphColoringUtil(graph, m, color, v + 1, V)) {

return true;

}

// If assigning color c doesn't lead to a solution, then remove it

color[v] = 0;

}

}

return false; // No color can be assigned to this vertex

}

// Function to solve the graph coloring problem

bool graphColoring(bool graph[][MAX\_VERTICES], int m, int V) {

int color[MAX\_VERTICES] = {0}; // Initialize all vertices as uncolored

if (!graphColoringUtil(graph, m, color, 0, V)) {

printf("Solution does not exist\n");

return false;

}

printf("Solution exists with following color assignments:\n");

printSolution(color, V);

return true;

}

// Main function

int main() {

int V, m;

printf("Enter the number of vertices in the graph: ");

scanf("%d", &V);

printf("Enter the adjacency matrix (0/1):\n");

bool graph[MAX\_VERTICES][MAX\_VERTICES];

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

for (int j = 0; j < V; j++) {

scanf("%d", &graph[i][j]);

}

}

printf("Enter the number of colors: ");

scanf("%d", &m);

graphColoring(graph, m, V);

return 0;

}

**Output:**

Enter the number of vertices in the graph: 4

Enter the adjacency matrix (0/1):

0 1 1 1

1 0 1 0

1 1 0 1

1 0 1 0

Enter the number of colors: 4

Solution exists with following color assignments:

Vertex Color

0 1

1 2

2 3

3 2

--------------------------------

Process exited after 27.81 seconds with return value 0

Press any key to continue . . .

