PRACTICAL 8

Name: Ishita Premchandani
Roll no.: A3_B1_08

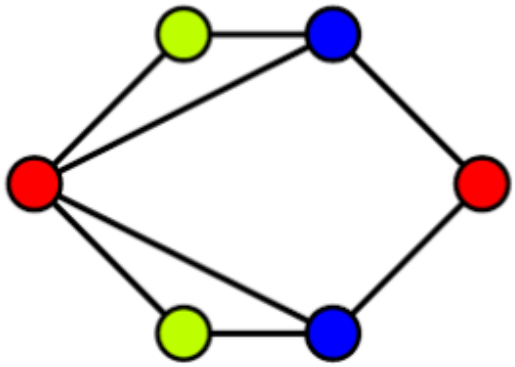
Aim: Implement Graph Colouring algorithm use Graph colouring concept.
Problem Statement:
A GSM is a cellular network with its entire geographical range divided into hexadecimal cells. Each cell has a communication tower which connects with mobile phones within cell. Assume this GSM network operates in different frequency ranges. Allot frequencies to each cell such that no adjacent cells have same frequency range.
Consider an undirected graph G = (V, E) shown in fig. Find the colour assigned to each node using Backtracking method. Input is the adjacency matrix of a graph G(V, E), where V is the number of Vertices and E is the number of edges.

Code:

```
#include <stdio.h>

int isSafe(int v, int graph[20][20], int color[], int c, int V) {
    for (int i = 0; i < V; i++)
        if (graph[v][i] && color[i] == c)
            return 0;
    return 1;
}

int solve(int graph[20][20], int m, int color[], int v, int V) {
    if (v == V)
        return 1;

    for (int c = 1; c <= m; c++) {
        if (isSafe(v, graph, color, c, V)) {
            color[v] = c;
            if (solve(graph, m, color, v + 1, V))
```

```
            return 1;
        color[v] = 0;
    }
  }
  return 0;
}

int main() {
  int V, m;
  int graph[20][20], color[20] = {0};

  printf("Enter number of vertices: ");
  scanf("%d", &V);

  printf("Enter adjacency matrix:\n");
  for (int i = 0; i < V; i++)
    for (int j = 0; j < V; j++)
      scanf("%d", &graph[i][j]);

  printf("Enter number of colors: ");
  scanf("%d", &m);

  if (!solve(graph, m, color, 0, V)) {
    printf("No solution\n");
    return 0;
  }

  printf("Assigned Colors:\n");
  for (int i = 0; i < V; i++)
    printf("Vertex %d -> Color %d\n", i, color[i]);

  return 0;
}
```

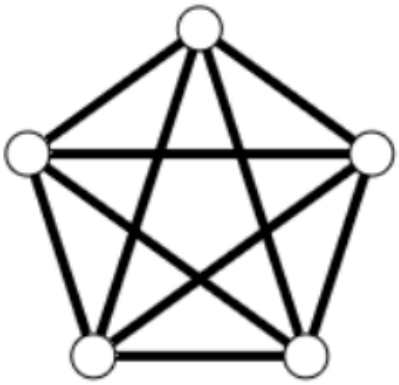
Graph 1:



Output:

```
Output                                                    Clear
Enter number of vertices: 6
Enter adjacency matrix:
0 1 1 0 1 1
1 0 1 0 0 0
1 1 0 1 0 0
0 0 1 0 1 0
1 0 0 1 0 1
1 0 0 0 1 0
Enter number of colors: 3
Assigned Colors:
Vertex 0 -> Color 1
Vertex 1 -> Color 2
Vertex 2 -> Color 3
Vertex 3 -> Color 1
Vertex 4 -> Color 2
Vertex 5 -> Color 3


=== Code Execution Successful ===
```

Graph 2:



main.c Output

```
Enter number of vertices: 5
Enter adjacency matrix:
0 1 1 1 1
1 0 1 1 1
1 1 0 1 1
1 1 1 0 1
1 1 1 1 0
Enter number of colors: 1
No solution
```