Name- Anjali Lanjewar

Section- A4-B2

Roll no- 25

PRACTICAL NO. 8

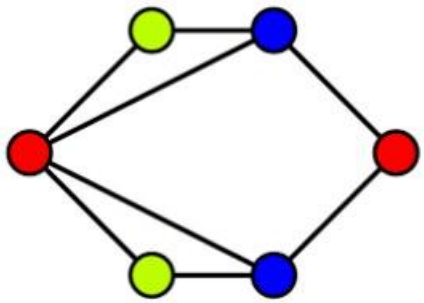
Aim: Implement Graph Colouring algorithm use Graph colouring concept.

Problem Statement:

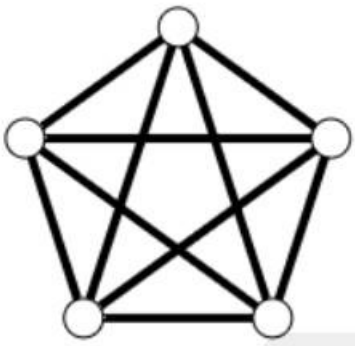
A GSM is a cellular network with its entire geographical range divided into hexadecimal cells. Each cell has a communication tower which connects with mobile phones within cell. Assume this GSM network operates in different frequency ranges. Allot frequencies to each cell such that no adjacent cells 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.

Graph 1:



Graph 2:



C code-

```
#include <stdio.h>

#define V 5

void printSolution(int color[], int n) {
printf("A valid coloring is:\n");     for
(int i = 0; i < n; i++) {
      printf("  Cell %d (Vertex %d) ---> Frequency %d (Color %d)\n", i, i,
color[i], color[i]);
   }
}

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

```
int graphColoringUtil(int graph[][V], int m, int color[], int v, int n) {
   if (v == n) {
return 1;
   }

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

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

         color[v] = c;

         if (graphColoringUtil(graph, m, color, v + 1, n) == 1) {
return 1;
         }

         color[v] = 0;
      }
   }

   return 0;
}
```

```
void solveGraphColoring(int graph[][V], int n, const char*
graphName) {
   printf("--- Analyzing %s ---\n", graphName);


   int color[V] = {0};


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


      for(int i = 0; i < n; i++) {
color[i] = 0;
      }


      if (graphColoringUtil(graph, m, color, 0, n) == 1) {
printf("Problem Solved.\n");
         printf("The Chromatic Number (minimum frequencies needed)
is: %d\n", m);
         printSolution(color, n);
printf("\n");          return;
      }
   }


   printf("No solution exists (this should not happen if m=V).\n\n");
}
```

```
int main() {     int
graph1[V][V] = {
        {0, 1, 1, 0, 0},
        {1, 0, 0, 1, 0},
        {1, 0, 0, 0, 1},
        {0, 1, 0, 0, 1},
        {0, 0, 1, 1, 0}
    };

    int graph2[V][V] = {
        {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}
    };


printf("================================================
\n");
    printf("Graph Coloring (Backtracking) for GSM Frequencies\n");

printf("================================================
```

```
\n\n");

    solveGraphColoring(graph1, V, "Graph 1 (5-Cycle)");
solveGraphColoring(graph2, V, "Graph 2 (Complete Graph K5)");

    return 0;
}
```

Output-

Output

```
==================================================
Graph Coloring (Backtracking) for GSM Frequencies
==================================================

--- Analyzing Graph 1 (5-Cycle) ---
Problem Solved.
The Chromatic Number (minimum frequencies needed) is: 3
A valid coloring is:
  Cell 0 (Vertex 0) ---> Frequency 1 (Color 1)
  Cell 1 (Vertex 1) ---> Frequency 2 (Color 2)
  Cell 2 (Vertex 2) ---> Frequency 2 (Color 2)
  Cell 3 (Vertex 3) ---> Frequency 1 (Color 1)
  Cell 4 (Vertex 4) ---> Frequency 3 (Color 3)

--- Analyzing Graph 2 (Complete Graph K5) ---
Problem Solved.
The Chromatic Number (minimum frequencies needed) is: 5
A valid coloring is:
  Cell 0 (Vertex 0) ---> Frequency 1 (Color 1)
  Cell 1 (Vertex 1) ---> Frequency 2 (Color 2)
  Cell 2 (Vertex 2) ---> Frequency 3 (Color 3)
  Cell 3 (Vertex 3) ---> Frequency 4 (Color 4)
  Cell 4 (Vertex 4) ---> Frequency 5 (Color 5)



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