Practical no: 08

Name: Prarthana Agrawal
Class: A4_B1
Roll no: 15

Aim: Implement Graph Colouring algorithm use Graph colouring concept.

Problem Statement:

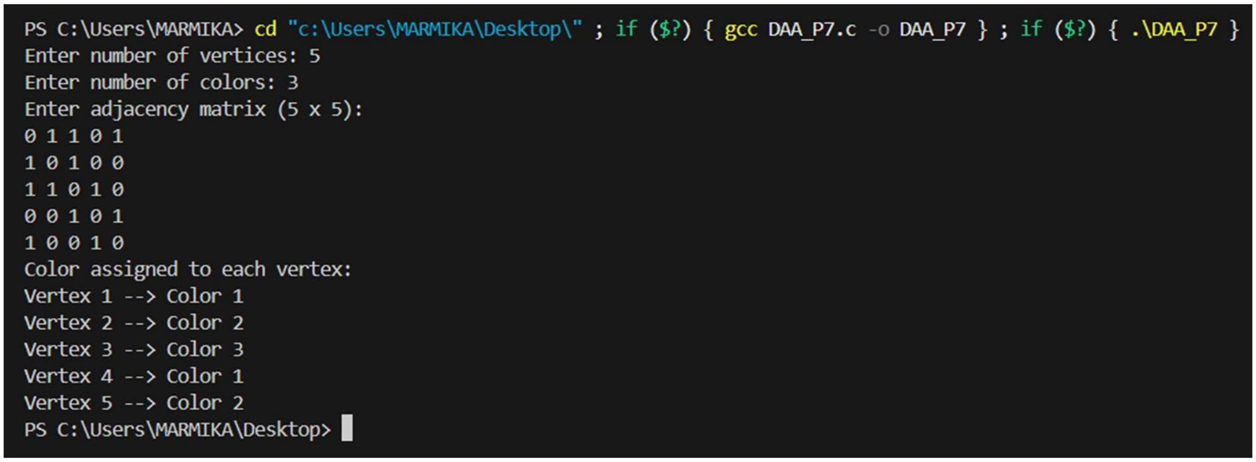
A GSM is a cellular network with its entire geographical range divided into hexadecimal cells. Each cell has a communication tower which connects with mobile phones within cell. Assume this GSM network operates in different frequency ranges. Allot frequencies to each cell such that no adjacent cells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

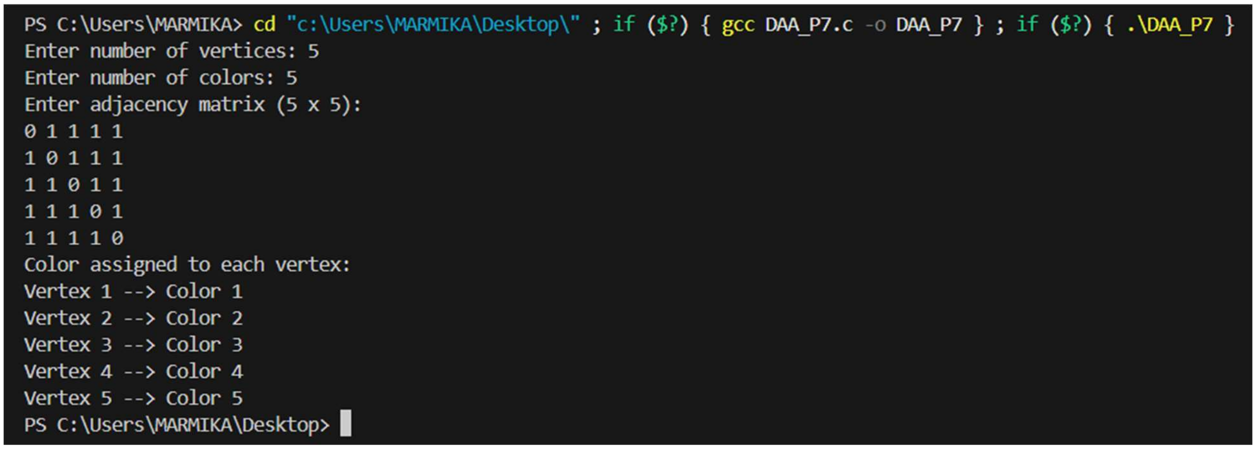
CODE-

```
#include <stdio.h>
#include <stdbool.h>

#define MAX 20

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

bool graphColoringUtil(int graph[MAX][MAX], int m, int color[], int v, int V) {
        if (v == V)
        return true;

        for (int c = 1; c <= m; c++) {
        if (isSafe(v, graph, color, c, V)) {
        color[v] = c;
        if (graphColoringUtil(graph, m, color, v + 1, V))
                return true;
        color[v] = 0;
        }
        }
        return false;
}

void graphColoring(int graph[MAX][MAX], int m, int V) {
        int color[MAX];
        for (int i = 0; i < V; i++)
        color[i] = 0;

        if (!graphColoringUtil(graph, m, color, 0, V)) {
      printf("No solution exists using %d colors.\n", m);
        return;
        }

        printf("Color assigned to each vertex:\n");
        for (int i = 0; i < V; i++)
      printf("Vertex %d --> Color %d\n", i + 1, color[i]);
}

int main() {
        int V, m;
        int graph[MAX][MAX];

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

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

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

   graphColoring(graph, m, V);
        return 0;
}
```

OUTPUT-



```
PS C:\Users\MARMIKA> cd "c:\Users\MARMIKA\Desktop\" ; if ($?) { gcc DAA_P7.c -o DAA_P7 } ; if ($?) { .\DAA_P7 }
Enter number of vertices: 5
Enter number of colors: 3
Enter adjacency matrix (5 x 5):
0 1 1 0 1
1 0 1 0 0
1 1 0 1 0
0 0 1 0 1
1 0 0 1 0
Color assigned to each vertex:
Vertex 1 --> Color 1
Vertex 2 --> Color 2
Vertex 3 --> Color 3
Vertex 4 --> Color 1
Vertex 5 --> Color 2
PS C:\Users\MARMIKA\Desktop>
```



```
PS C:\Users\MARMIKA> cd "c:\Users\MARMIKA\Desktop\" ; if ($?) { gcc DAA_P7.c -o DAA_P7 } ; if ($?) { .\DAA_P7 }
Enter number of vertices: 5
Enter number of colors: 5
Enter adjacency matrix (5 x 5):
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
Color assigned to each vertex:
Vertex 1 --> Color 1
Vertex 2 --> Color 2
Vertex 3 --> Color 3
Vertex 4 --> Color 4
Vertex 5 --> Color 5
PS C:\Users\MARMIKA\Desktop>
```