DAA Assignment 5

Name: Digvijay Pawar
Class: TY.Btech Comp B2
Gr.No: 21810344
Roll No: 322043

Backtracking Implementation (Graph coloring)

Code Implementation :

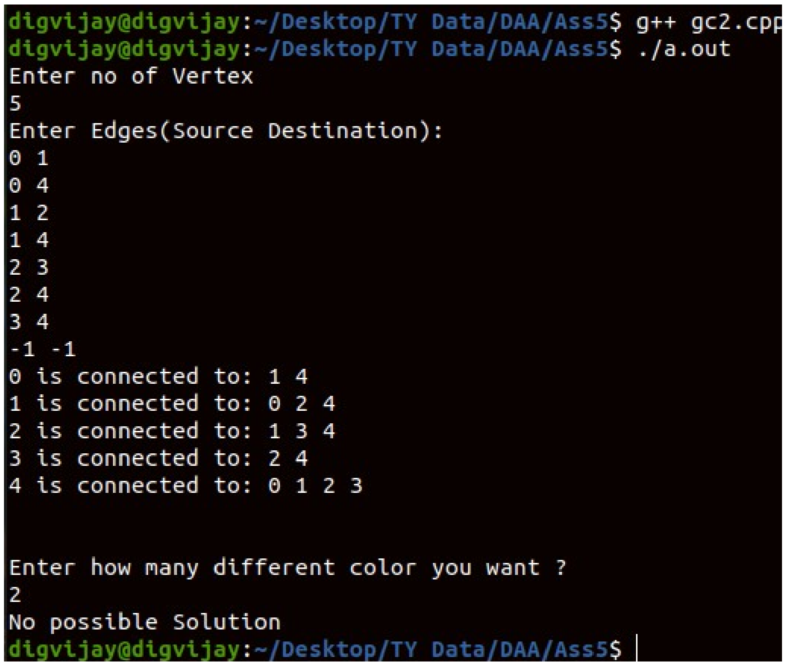
```
#include <iostream>
#include <vector>
using namespace std;
int sol=1,flag=0;
struct Edge {
	int src, dest;
};
class Graph
{
  public:
	vector<vector<int>> adj;
	Graph(vector<Edge> &edges, int N)
	{
		adj.resize(N);
		for (Edge edge: edges)
		{
			int src = edge.src;
			int dest = edge.dest;
```

```
                        adj[src].push_back(dest);
                        adj[dest].push_back(src);
                }
        for (int i = 0; i < adj.size(); i++) {
          std::cout <<i<<" is connected to: ";
            for (int j = 0; j < adj[i].size(); j++)
                cout << adj[i][j] << " ";
            cout << endl;
        }
        std::cout<< '\n';
           }
};
string COLORS[] = {"", "BLUE", "GREEN", "RED", "YELLOW", "ORANGE",
                                "PINK", "BLACK", "BROWN", "WHITE", "PURPLE"};
bool isSafe(Graph &graph, vector<int> color, int v, int c)
{
        for (int u : graph.adj[v])
                if (color[u] == c)
                        return false;
        return true;
}
void colorable(Graph &graph, vector<int> &color, int k, int v, int N)
{
        if (v == N)
        {
                 flag=1;
                std::cout <<"Solution "<<sol<<": ";
                for (int v = 0; v < N; v++)
                {
                cout<< COLORS[color[v]] <<" ";
                }
                cout << endl;
                 sol++;
                return;
        }

        for (int c = 1; c <= k; c++)
        {
                if (isSafe(graph, color, v, c))
                {
                        color[v] = c;
                        colorable(graph, color, k, v + 1, N);
                        color[v] = 0;
                }
        }
}
int main()
{
  int N;
        std::cout << "Enter no of Vertex" << '\n';
  std::cin >> N;
  std::cout << "Enter Edges(Source Destination):" << '\n';
        vector<Edge> edges;
  int a,b;
  while(a!=-1){
    cin>>a>>b;
    if (a==-1)
      break;
    edges.push_back({a,b});
  }
  Graph g(edges, N);
  int k;
  std::cout << "\nEnter how many different color you want ?" << '\n';
  std::cin >>k;
        vector<int> color(N, 0);
        colorable(g, color, k, 0, N);
        if(flag==0)
```

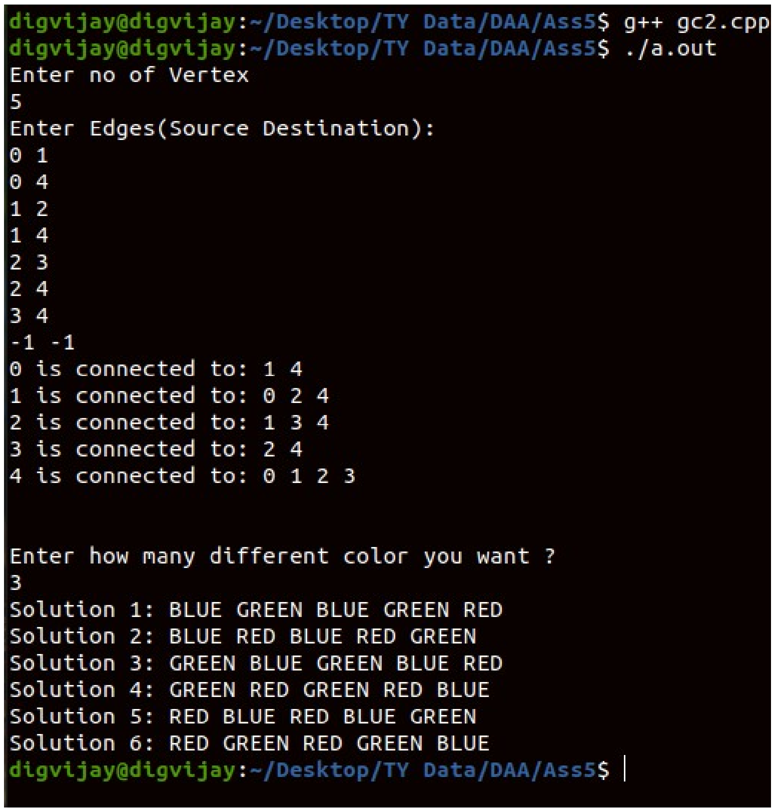
```
            std::cout << "No possible Solution" << '\n';
    return 0;
}
```

Output:



```
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$ g++ gc2.cpp
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$ ./a.out
Enter no of Vertex
5
Enter Edges(Source Destination):
0 1
0 4
1 2
1 4
2 3
2 4
3 4
-1 -1
0 is connected to: 1 4
1 is connected to: 0 2 4
2 is connected to: 1 3 4
3 is connected to: 2 4
4 is connected to: 0 1 2 3


Enter how many different color you want ?
2
No possible Solution
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$
```



```
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$ g++ gc2.cpp
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$ ./a.out
Enter no of Vertex
5
Enter Edges(Source Destination):
0 1
0 4
1 2
1 4
2 3
2 4
3 4
-1 -1
0 is connected to: 1 4
1 is connected to: 0 2 4
2 is connected to: 1 3 4
3 is connected to: 2 4
4 is connected to: 0 1 2 3


Enter how many different color you want ?
3
Solution 1: BLUE GREEN BLUE GREEN RED
Solution 2: BLUE RED BLUE RED GREEN
Solution 3: GREEN BLUE GREEN BLUE RED
Solution 4: GREEN RED GREEN RED BLUE
Solution 5: RED BLUE RED BLUE GREEN
Solution 6: RED GREEN RED GREEN BLUE
digvijay@digvijay:~/Desktop/TY Data/DAA/Ass5$
```