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);

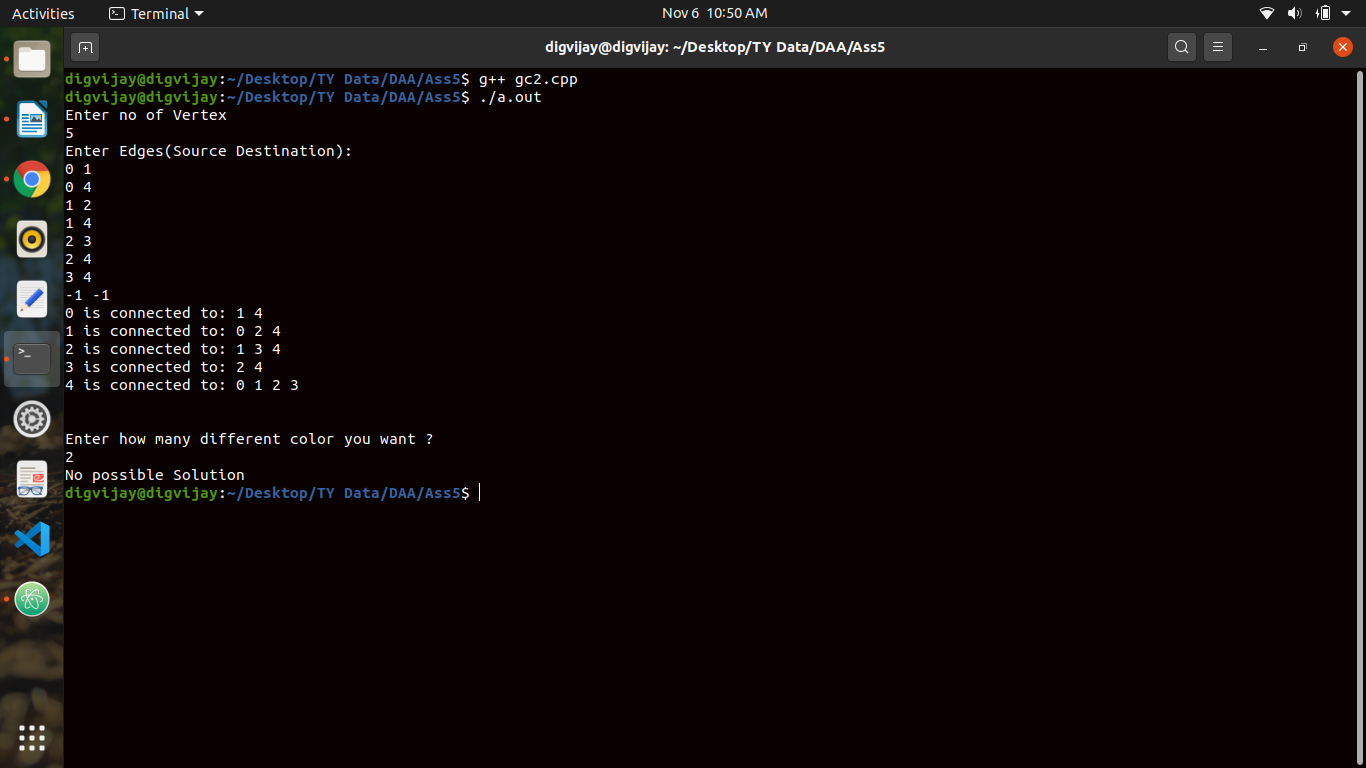
colorable(g, color, k, 0, N);

if(flag==0)

std::cout << "No possible Solution" << '\n';

return 0;

}

***Output:***

