

Talent Battle 100 Days Coding Series

Given an undirected graph and an integer M . The task is to determine if the graph can be colored with at most M colors such that no two adjacent vertices of the graph are colored with the same color. Here coloring of a graph means the assignment of colors to all vertices. Print 1 if it is possible to colour vertices and 0 otherwise.

Example 1:

Input:

$N = 4$

$M = 3$

$E = 5$

Edges[] = {(0,1),(1,2),(2,3),(3,0),(0,2)}

Output: 1

Explanation: It is possible to colour the given graph using 3 colours.

Example 2:

Input:

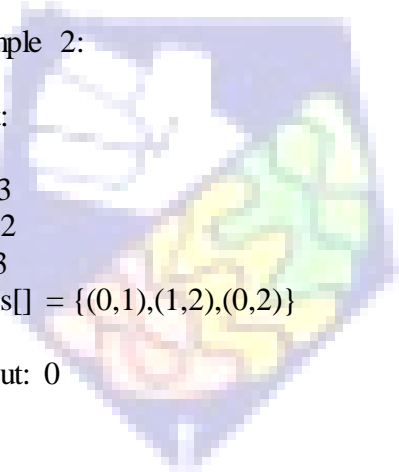
$N = 3$

$M = 2$

$E = 3$

Edges[] = {(0,1),(1,2),(0,2)}

Output: 0



TalentBattle

C++

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
#define ll long long
```

```
int main()
```

```
{
```

```
    ios_base::sync_with_stdio(0);
```

```
    cin.tie(0); cout.tie(0);
```

```
    ll n,m,e;
```

```
        cin>>n>>m>>e;
```

```
        vector<int>adj[n];
```

```
        for(int i=0;i<e;i++)
```

```
        {
```

```
            int a,b;
```

```
            cin>>a>>b;
```

```
            adj[a].push_back(b);
```

```
            adj[b].push_back(a);
```

```
        }
```

```
        vector<int>vis(n,0);
```

```
        vector<int>color(n,1);
```

```
        int maxicolor=1;
```

```
        for(int i=0;i<n;i++)
```

```
        {
```

```
            if(!vis[i])
```

```
            {
```

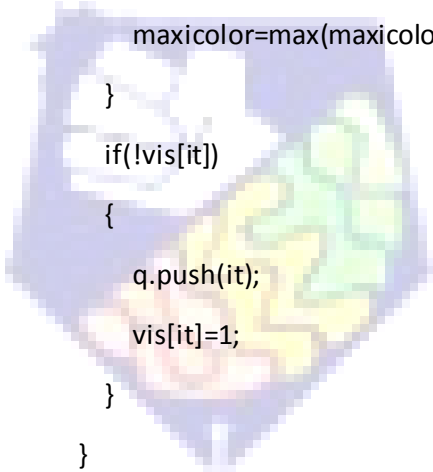
```
                vis[i]=1;
```

TalentBattle

Talent Battle 100 Days Coding Series

```
queue<int>q;
q.push(i);
while(!q.empty())
{
    int front=q.front();
    q.pop();
    for(auto it:adj[front])
    {
        if(color[it]==color[front])
        {
            color[it]++;
            maxicolor=max(maxicolor,color[it]);
        }
        if(!vis[it])
        {
            q.push(it);
            vis[it]=1;
        }
    }
}
}

if(maxicolor<=m)
    cout<<1<<"\n";
else
    cout<<0<<"\n";
}
```



TalentBattle

Talent Battle 100 Days Coding Series

Java

```
import java.util.*;
```

```
public class Main
```

```
{
```

```
    public boolean isPossible(boolean [][] graph,int []color,int node,int col,int n){
```

```
        for(int i=0;i<n;i++){
```

```
            if(graph[node][i] && color[i]==col)return false;
```

```
        }
```

```
        return true;
```

```
    }
```

```
    public boolean solve(int node,boolean [][] graph,int []color,int m,int n){
```

```
        if(node==n) return true;
```

```
        for(int i=1;i<=m;i++){
```

```
            if(isPossible(graph,color,node,i,n)){
```

```
                color[node]=i;
```

```
                if(solve(node+1,graph,color,m,n))return true;
```

```
                color[node]=0;
```

```
            }
```

```
        }
```

```
        return false;
```

```
    }
```

```
    public boolean graphColoring(boolean graph[][], int m, int n) {
```

```
        int []color= new int[n];
```

```
        return solve(0,graph,color,m,n);
```

```
    }
```

```
        public static void main(String[] args) {
```

Talent Battle 100 Days Coding Series

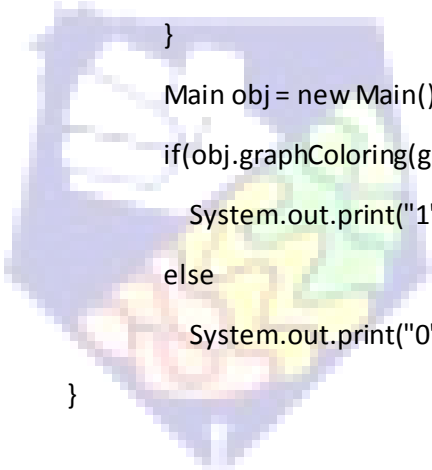
```
Scanner sc = new Scanner(System.in);

int n = sc.nextInt();
int m = sc.nextInt();
int e = sc.nextInt();
int a,b;

boolean graph[][] = new boolean[n][n];

for(int i=1 ; i<=e ; i++){
    a = sc.nextInt();
    b = sc.nextInt();
    graph[a][b] = true;
    graph[b][a] = true;
}

Main obj = new Main();
if(obj.graphColoring(graph,m,n))
    System.out.print("1");
else
    System.out.print("0");
}
}
```



TalentBattle

Talent Battle 100 Days Coding Series

Python

```
def isSafe(graph, color):  
    for i in range(n):  
        for j in range(i + 1, n):  
            if (graph[i][j] and color[j] == color[i]):  
                return False  
    return True
```

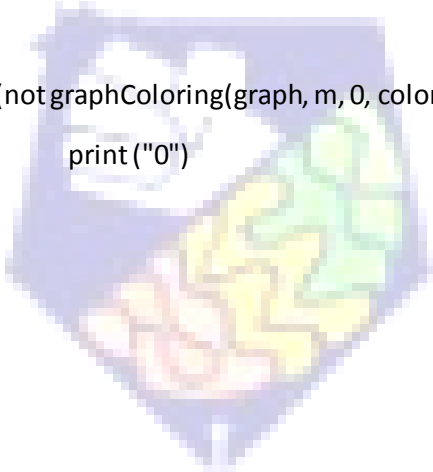
```
def graphColoring(graph, m, i, color):  
    if (i == n):  
        if (isSafe(graph, color)):  
            display(color)  
            return True  
        return False  
    for j in range(1, m + 1):  
        color[i] = j  
        if (graphColoring(graph, m, i + 1, color)):  
            return True  
    color[i] = 0  
    return False
```

```
def display(color):  
    print("1")
```

```
n=int(input())  
m=int(input())  
e=int(input())  
graph=[]  
for i in range(n):
```

Talent Battle 100 Days Coding Series

```
a=[]  
for j in range(n):  
    a.append(0)  
graph.append(a)  
for i in range(e):  
    a=int(input())  
    b=int(input())  
    graph[a][b]=1  
    graph[b][a]=1  
  
color= [0 for i in range(n)]  
  
if (not graphColoring(graph, m, 0, color)):  
    print("0")
```



TalentBattle