Answer to the question no-01

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int n;
   cin>>n;
   int A[n][n];
   for(int i=0;i<n;i++)
   {
      for(int j=0;j<n;j++)
      {
         cin>>A[i][j];
      }
   }
   for(int i=0;i<n;i++)
   {
      cout<<i<<": ";
      for(int j=0;j<n;j++)
      {
         if(A[i][j]==1)
         {
            cout<<j<<" ";
         }
      }
      cout<<endl;
   }
}
```

Answer to the question no-02

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5;
int visited[N];
vector<int>adj_list[N];
int level[N];
void BFS(int src)
```

```
{
   queue<int>q;
   visited[src]=1;
   level[N]=0;
   q.push(src);
   while(!q.empty())
   {
      int head=q.front();
      q.pop();
      for(int adj_node:adj_list[head])
      {
         if(visited[adj_node]==0)
         {
            visited[adj_node]=1;
            level[adj_node]=level[head]+1;
            q.push(adj_node);
         }
      }
   }
}
int main()
{
   int n,e;
   cin>>n>>e;
   for(int i=1;i<=e;i++)
   {
      int u,v;
      cin>>u>>v;
      adj_list[u].push_back(v);
      adj_list[v].push_back(u);
   }
   BFS(0);
   for(int i=0;i<n;i++)
   {
      cout<<"node "<<i<<"->"<<"level: "<<level[i]<<endl;;
   }
}
```

Answer to the question no-03

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5;
int visited[N];
vector<int>adj_list[N];
bool DFS(int node)
{
   visited[node]=1;
   for(int adj_node:adj_list[node])
   {
      if(visited[adj_node]==0)
      {
         bool run=DFS(adj_node);
         if(run)
         {
            return true;
         }
      }
      else if(visited[adj_node]==1)
      {
         return true;
      }
   }
   visited[node]=2;
   return false;
}
int main()
{
   int n,e;
   cin>>n>>e;
   for(int i=1;i<=e;i++)
   {
      int u,v;
      cin>>u>>v;
      adj_list[u].push_back(v);
```

```
    }
    for(int i=0;i<n;i++)
    {
        if(visited[i]==0)
        {
            bool run=DFS(i);
            if(run)
            {
                cout<<"YES\n";
                return 0;
            }
        }
    }
    cout<<"NO\n";
}
```

Answer to the question no-04

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5;
int visited[N];
vector<int>adj_list[N];
int color[N];
bool DFS(int node)
{
    visited[node]=1;
    for(int adj_node:adj_list[node])
    {
        if(visited[adj_node]==0)
        {
            if(color[node]==1)
            {
                color[adj_node]=2;
            }
            else
            {
                color[adj_node]=1;
```

```
            }
            bool is_bipartite=DFS(adj_node);
            if(!is_bipartite)
            {
                return false;
            }
        }
        else
        {
            if(color[node]==color[adj_node])
            {
                return false;
            }
        }
    }
    return true;
}
int main()
{
    int n,e;
    cin>>n>>e;
    for(int i=1; i<=e; i++)
    {
        int u,v;
        cin>>u>>v;
        adj_list[u].push_back(v);
        adj_list[v].push_back(u);
    }
    for(int i=0; i<n; i++)
    {
        if(visited[i]==0)
        {
            color[i]=1;
            bool run=DFS(i);
            if(!run)
            {
                cout<<"NO\n";
```

```
            return 0;
        }
      }
   }
   cout<<"YES\n";
}
```

Answer to the question no-05

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5;
int visited[N];
vector<int>adj_list[N];
void DFS(int node)
{
   visited[node]=1;
   for(int adj_node:adj_list[node])
   {
      if(visited[adj_node]==0)
      {
         DFS(adj_node);
      }
   }
}
int main()
{
   int n,e,cnt=0;
   cin>>n>>e;
   for(int i=1;i<=e;i++)
   {
      int u,v;
      cin>>u>>v;
      adj_list[u].push_back(v);
      adj_list[v].push_back(u);
   }
   for(int i=0;i<n;i++)
   {
      if(visited[i]==0)
```

```
        {
            DFS(i);
            cnt++;
        }
    }
    cout<<cnt<<endl;
}
```