```
//DFS
const int mx=30;
vector<int>adj[mx];
bool visited[mx];
void DFS(int source){
visited[source]=1;
for(auto node:adj[source]){
if(!visited[node]){
DFS(node);}}}
int main(){
int n,m;
cin>>n>>m;
int u,v;
for(int i=0;i<m;i++){
cin>>u>>v;
adj[u].push_back(v);
adj[v].push_back(u);}
DFS(1);
for(int i=1;i<=n;i++){
cout<<visited[i]<<" ";}
cout<<endl;}
//DFS ON GRID

const int mxi=1001;
int n, m;
int mx[mxi][mxi];
bool visited[mxi][mxi];
int sum=0;
int  DFS(int x, int y){
visited[x][y] = 1;
sum+=mx[x][y];
for (int i = 0; i < 4; i++){
int nx = x + dx[i];
int ny = y + dy[i];
if (nx >= 0 and nx <= n - 1 and ny >= 0 and ny <= m - 1 and
!visited[nx][ny] and mx[nx][ny]!=0){
DFS(nx, ny);}}
return sum;}
signed main(){
optimize();
int TC;
cin >> TC;
```

```
while (TC--){
cin >> n >> m;
for(int i=0;i<n;i++){
for(int j=0;j<m;j++){
visited[i][j]=0;}}
for (int i = 0; i < n; i++){
for (int j = 0; j < m; j++){
cin >> mx[i][j];}}
int ans=0;
for (int i = 0; i < n; i++){
for (int j = 0; j < m; j++){
if (!visited[i][j] and mx[i][j]!=0){
int sm=DFS(i, j);
ans=max(ans,sm);
sum=0;}}}
cout<<ans<<endl;}}
//MY TEMP
#include<bits/stdc++.h>
using namespace std;
#define int long long int
#define pb push_back
#define endl '\n'
#define mod 1000000007
const double eps = 1e-9;
const int inf = 2000000000;
const int infl = 9000000000000000000;
#define mem(a,b) memset(a, b, sizeof(a) )
#define gcd(a,b) __gcd(a,b)
#define optimize()
ios_base::sync_with_stdio(0);cin.tie(0);cout.tie(0);
int dx[] = {+1, 0, -1, 0, +1, +1, -1, -1};
int dy[] = {0, +1, 0, -1, +1, -1, +1, -1};
//BE
int BE(int x,int y){
int res=1;
while (y>0){
if(y%2==1){
res*=x;
res%=mod;}
x=(x*x)%mod;
y/=2;}
return res;}
```