**BFS (Ds cạnh)**

[**#include**](https://www.facebook.com/hashtag/include?__eep__=6&__cft__%5b0%5d=AZW0DGDlHRrSWHzS1MkQRRr_jYU5DtiNVSli7SVgI1zg1gjwdtgSHXr1SO8X-gkqbnK1RhPAiDz3SZ53dYsnPpkv3XaIKrI5cumzYZQy6B0R7sJhR470-GmoFYv4vWEyxodgDmepuXb4UazG8n3jHiWDyrt_NlaFJOx_aaxyoYSSS0s_attes1iUOY7TT32rWoc&__tn__=R%5d-R)<stdio.h>

int Q[100],d=0,c=0;

int A[100][100],N,M,chuaxet[100];

FILE \*f1 = fopen("DSCANH.IN","r");

FILE \*f2 = fopen("BFS.OUT","w");

void DocDSCanh(){

fscanf(f1,"%d %d",&N,&M);

int i,j,k;

for(k=1;k<=M;k++){

fscanf(f1,"%d%d",&i,&j);

A[i][j]=1; A[j][i]=1;

}

fclose(f1);

}

void push(int x){//thêm x vào Q

Q[d]=x; d++;

}

int empty(){

return (c>d);

}

int pop(){//bỏ phần tử vị trí Q[c]

int x = Q[c];

c++;

return x;

}

void BFS(int i){

int x,j;

chuaxet[i]=0;

push(i);//i

while(!empty()){

x = pop();

if(!empty())fprintf(f2,"%d ",x);

for(j=1;j<=N;j++){

if(A[x][j]&&chuaxet[j]){

push(j);

chuaxet[j]=0;

}

}

}

}

main(){

int i;

DocDSCanh();

for(i=1;i<=N;i++) chuaxet[i]=1;

for(i=1;i<=N;i++)

if(chuaxet[i])

BFS(i);

fclose(f2);

}

**DFS(Ds cạnh)**

[**#include**](https://www.facebook.com/hashtag/include?__eep__=6&__cft__%5b0%5d=AZVFKQ3VNYkmbPX_5AT4NMmZygDNY8GEyv5w7GVfvJJPmBZqWlqbV3P4y36Ht6LAVe0VcOMc4CfUliBdYz3k9LEG-U7zVvBeP_KhU3umigp0jSAr4l3X1XVhwRahA6CP1kcFiRy2ApOqZmPlkj0V_i6KTADn-royCVPGaj-xBe2dTegFpwN_CTQEWoeN68EKD5M&__tn__=R%5d-R)<stdio.h>

FILE \*f1 = fopen("DSCANH.IN","r");

FILE \*f2 = fopen("DFS.OUT","w");

int A[100][100],N,M,chuaxet[100];

void Init(){

for(int i=1;i<=N;i++)

chuaxet[i]=1;

}

void DocDSCanh(){

fscanf(f1,"%d %d",&N,&M);

int i,j,k;

for(k=1;k<=M;k++){

fscanf(f1,"%d%d",&i,&j);

A[i][j]=1; A[j][i]=1;

}

fclose(f1);

}

void DFS(int i){

fprintf(f2,"%d ",i);

chuaxet[i]=0;

for(int j=1;j<=N;j++)

if(chuaxet[j]&&A[i][j])

DFS(j);

}

main(){

int i;

DocDSCanh();

Init();

int t,u;

fscanf(f1,"%d",&t);

for(i=1;i<=N;i++)

if(chuaxet[i])

DFS(i);

fclose(f2);

}

**Tree-BFS**

#include <iostream>

#include <fstream>

[**#include**](https://www.facebook.com/hashtag/include?__eep__=6&__cft__%5b0%5d=AZUNFkDij5f895FDMk6p9HgtegUl-TT0svgE1su0UQZv1YTDOEyRvMfr90G6FhKWAx2NWqZm3Z8f6-MUibn13ZwEDq9kInAMHoSUo0OkVvS017lkQ_fqEZ-rrAUIKAJpujjxmy4niRICLZGMm5PXwtSH-8fcnLhcoJPoLAjU-xeoXjoYHRPA7XSjHK37HDKO0VU&__tn__=R%5d-R) <queue>

using namespace std;

int N, K;

int arr[100][100];

int visited[100];

ifstream inp("Data.in");

ofstream out("Data.out");

struct Edge

{

int p1;

int p2;

};

Edge edgeArr[100];

int rsSize = 0;

void BFS(int j)

{

queue<int> q;

q.push(j);

visited[j] = 1;

while (!q.empty())

{

int e = q.front();

q.pop();

for (int i = 1; i <= N; i++)

{

if (arr[e][i] == 1 && visited[i] == 0)

{

Edge t;

t.p1 = e > i ? i : e;

t.p2 = e > i ? e : i;

edgeArr[rsSize++] = t;

q.push(i);

visited[i] = 1;

}

}

}

cout << endl;

}

void Init()

{

for (int i = 1; i <= N; i++)

{

visited[i] = 0;

}

}

int main()

{

inp >> N;

inp >> K;

string t;

getline(inp, t);

for (int i = 1; i <= N; i++)

{

string tmp;

getline(inp, tmp);

for (int j = 0; j < tmp.length(); j++)

{

if (tmp.at(j) != ' ')

{

arr[i][tmp.at(j) - '0'] = 1;

arr[tmp.at(j) - '0'][i] = 1;

}

}

}

BFS(K);

out << N << " " << rsSize << endl;

for (int i = 0; i < rsSize; i++)

{

out << edgeArr[i].p1 << " " << edgeArr[i].p2 << endl;

}

cout << "Done!!" << endl;

}

**Tree-DFS**

#include <iostream>

#include <fstream>

[**#include**](https://www.facebook.com/hashtag/include?__eep__=6&__cft__%5b0%5d=AZVWaSSwQ0QKF-J_zhR8-KBBNyPXeKp85iZD5ztLd689R0KlV_oj1ORYP2-pSZ50fVqmIDstvt7GWC6HeH26AMpntGf8YFtAfCKKxuHwGNy3GKSiO4wFa7kxcs6VxiVnr7l8WDNCoAkUTtea-aR-UXnDWViIfwwwheAHcNXHuv4_wU6Dh61jEh4L3kmANlSH9H0&__tn__=R%5d-R) <stack>

using namespace std;

int N, K;

int arr[100][100];

int visited[100];

ifstream inp("Data.in");

ofstream out("Data.out");

struct Edge

{

int p1;

int p2;

};

Edge edgeArr[100];

int rsSize = 0;

void DFS(int i)

{

stack<int> s;

s.push(i);

visited[i] = 1;

while (!s.empty())

{

int e = s.top();

s.pop();

for (int i = 1; i <= N; i++)

{

if (arr[e][i] == 1 && visited[i] == 0)

{

Edge t;

t.p1 = e > i ? i : e;

t.p2 = e > i ? e : i;

edgeArr[rsSize++] = t;

s.push(e);

s.push(i);

visited[i] = 1;

break;

}

}

}

}

void Init()

{

for (int i = 1; i <= N; i++)

{

visited[i] = 0;

}

}

int main()

{

inp >> N;

inp >> K;

string t;

getline(inp, t);

for (int i = 1; i <= N; i++)

{

string tmp;

getline(inp, tmp);

for (int j = 0; j < tmp.length(); j++)

{

if (tmp.at(j) != ' ')

{

arr[i][tmp.at(j) - '0'] = 1;

arr[tmp.at(j) - '0'][i] = 1;

}

}

}

DFS(K);

out << N << " " << rsSize << endl;

for (int i = 0; i < rsSize; i++)

{

out << edgeArr[i].p1 << " " << edgeArr[i].p2 << endl;

}

}