一、

A A B

二、

1.

（1）V1 V2 V4 V8 V5 V3 V6 V7

（2）V1 V2 V3 V4 V5 V6 V7 V8

（3）<V1,V3> <V3,V6> <V3,V7> <V1,V2> <V2,V5> <V2,V4> <V5,V8>

2.

A C B D E

A C D B E

A C D E B

A D C B E

A D C E B

4.

边：统计邻接矩阵不为0(或INF)的总数并除以二。

i，j之间是否有边相连：判断A[i,j]是否为0(或INF)

判断i，j是否连通:从i行开始递推存在边的点看是否能推到(x,j)

顶点的度：统计第i列不为0(或INF)的值的总数

5.

dis[];

A->B dis[B] = 10;

A->C dis[C] = 18;

B->D dis[D] = 10+5 = 15;

C->B dis[B] = min(10,17+5)=10;

D->C dis[C] = min(17 , 18) = 17;

D->E dis[E] = 15+2 = 17;

E->C dis[C] = min(17+2 , 17)=17;

B: 10 A->B

C: 17 A->B->D->C

D: 15 A->B->D

E: 17 A->B->D->E

3.

#include<cstdio>

#include<iostream>

using namespace std;

const int maxn = 10005;

struct edge{

int v,w,next;

}e[maxn];

int k,head[maxn];

void adde(int u, int v ,int w){

e[k].v=v;

e[k].w=w;

e[k].next=head[u];

head[u]=k++;

}

void matrixTOlist(){

k=0;

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

for(int j=i+1 ;j<=n ;j++){

if(a[i][j]!=INF)

adde(i,j,a[i][j]);

adde(j,i,a[j][i]);

}

}

bool haveRoot(){

for(int i=1 ;i<=n ;i++){

bool flag = 0;

for(int j=1;j<=n;j++){

if(a[i][j] == INF){

flag = 1;

break;

}

}

if(!flag)

return true;

}

return false;

}

int main(){

matrixTOlist();

bool havert = haveRoot();

return 0;

}