/\*

You have a business with several offices;

you want to lease phone lines to connect them up with each other;

and the phone company charges different amounts of money to connect different pairs of cities.

You want a set of lines that connects all your offices with a minimum total cost.

Solve the problem by suggesting appropriate data structures.

\*/

#include<iostream>

#include<stdio.h>

using namespace std;

int weight[10][10],nodes,visited[10];

int main()

{

int i,j,k,l,m,min,total\_cost=0;

for(i=0;i<nodes;i++)

{

visited[i]=0;

}

cout<<"Enter total no. of nodes:";

cin>>nodes;

for(i=0;i<nodes;i++)

{

cout<<"\nEnter Values for Row "<<i+1<<"\t";

for(j=0;j<nodes;j++)

{

cin>>weight[i][j];

}

}

cout<<"Printing the Matrix\n";

for(i=0;i<nodes;i++)

{

cout<<"\nRow "<<i+1<<"\n";

for(j=0;j<nodes;j++)

{

cout<<weight[i][j]<<"\t";

}

}

min=9999;

//find the edge having minimum weight from the entire graph

for(i=0;i<nodes;i++)

{

for(j=0;j<nodes;j++)

{

if(weight[i][j]<min&&weight[i][j]!=0)

{

min=weight[i][j];

k=i;

l=j;

}

}

}

printf("\nEdge %d-%d having weight %d",k,l,min);

visited[k]=1;

visited[l]=1;

total\_cost=min;

for(m=0;m<nodes-2;m++)

{

min=9999;

for(i=0;i<nodes;i++)

{

if(visited[i]==1)

{

for(j=0;j<nodes;j++)

{

if(visited[j]!=1)

{

if(weight[i][j]<min&&weight[i][j]!=0)

{

min=weight[i][j];

k=i;

l=j;

}

}

}

}

}

printf("\nEdge %d-%d having weight %d",k,l,min);

visited[k]=1;

visited[l]=1;

total\_cost=total\_cost+min;

}

printf("\ntotal cost is %d",total\_cost);

//getch();

return 0;

}

/\*

Enter total no. of nodes:2

Enter Values for Row 1 1

2

Enter Values for Row 2 2

1

Printing the Matrix

Row 1

1 2

Row 2

2 1

Edge 0-0 having weight 1

total cost is 1

\*/