#include<iostream>

#define SIZE 20

#define INFINITY 99

using namespace std;

class MST {

private :

int G [SIZE ][SIZE ],nodes ;

public :

MST();

void Prim ();

void get\_data();

};

MST ::MST (){

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

{

for (int j = 0; j < SIZE; j++)

{

G[i][j]=0;

}

}

}

void MST :: Prim ()

{

int select [SIZE ],i,j,k;

int min\_dist ,v1 ,v2 ,total = 0;

cout<<"\nThe minimum spanning tree "<<endl;

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

{

select[i]= 0;

select[0]=1;

}

for (int k = 0; k <nodes-1 ; k++)

{

min\_dist = INFINITY;

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

{

for (int j = 0; j <nodes-1; j++)

{

if (G[i][j] && ((select[i]&& !select[j] )||(!select [i ]&& select [j])))

{

if (G[i][j]<min\_dist )

{

min\_dist = G[i][j];

v1 = i;

v2 = j;

}

}

}

}

cout<<"Edge "<<v1<<" " <<v2<< " and Weight = "<<min\_dist<<"\n";

select [v1 ]= select[v2]= 1;

total= total +min\_dist ;

}

cout<<"\nTotal path length is "<<total<<endl;

}

void MST::get\_data(){

int v1 ,v2 ,length ,n;

cout <<"\nEnter the total number of nodes in the graph : ";

cin>>nodes;

cout <<"\nEnter the total number of edges in the graph : ";

cin>>n;

cout<<"\nEnter the edge and weight of the graph : ";

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

{

cout<<"\nEnter the edge in the form of v1 , v2 (for e.g. if there is edge between 0 and 1 then write 0 1) "<<endl;

cin>>v1>>v2;

cout<<"\nEnter corresponding weight : "<<endl;

cin>>length;

G[v1][v2]= G[v2][v1]= length;

}

}

int main (){

MST obj ;

cout<<"\nPrim's Algorithm\n";

obj.get\_data();

cout<<" "<<endl;

obj.Prim();

return 0 ;

}

/\*

\*/