/\*

***Name* :-** Abrar Mahedavi

***Roll No.* :-** 21035

***Title* :-** A newspaper delivery boy everyday drops newspaper in a society having many lane and each lane have many houses. Design a program to provide different paths that he could follow and suggest he path which will take him to finish his task with less efforts. Solve the problem by suggesting appropriate data structures. Design necessary class.

/\*

***Program :***

#include<iostream>

using namespace std;

#define Infinity 999

#define MAX 20

class graph

{

private:

int graph1[MAX][MAX];

int span\_tree[MAX][MAX];

int v;

public:

void input();

void display();

void prims();

};

void graph::input()

{

int i,j,wt;

cout<<"\nEnter the number of vertices:\t";

cin>>v;

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

{

for(j=1;j<=v;j++)

{

graph1[i][j]=Infinity;

}

}

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

{

for(j=i+1;j<=v;j++)

{

cout<<"\nEnter the weights between"<< i<<" and"<< j<<":\t";

cin>>wt;

graph1[i][j]=graph1[j][i]=wt;

}

}

}

void graph::display()

{

int i,j;

cout<<"\nThe matrix representation is:\n";

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

{

for(j=1;j<=v;j++)

{

cout<<"\t"<<graph1[i][j];

}

cout<<"\n";

}

cout<<"\nThe spanning tree is:\n";

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

{

for(j=1;j<=v;j++)

{

cout<<"\t"<<span\_tree[i][j];

}

cout<<"\n";

}

}

void graph::prims()

{

int i,j,u,ver;

int dist[MAX],visit[MAX],cost[MAX][MAX],via[MAX];

int min\_dist,edges=0,min\_cost=0;;

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

visit[i]=0;

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

{

for(j=1;j<=v;j++)

{

span\_tree[i][j]=0;

}

}

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

{

for(j=1;j<=v;j++)

{

if(graph1[i][j]==0)

cost[i][j]=Infinity;

else

cost[i][j]=graph1[i][j];

}

}

visit[1]=1;

dist[1]=0;

via[1]=1;

for(i=2;i<=v;i++)

{

dist[i]=cost[1][i];

via[i]=1;

}

while(edges< v-1)

{

min\_dist=Infinity;

for(i=2;i<=v;i++)

if(visit[i]==0&&dist[i]<min\_dist)

{

min\_dist=dist[i];

ver=i;

}

u=via[ver];

span\_tree[u][ver]=dist[ver];

span\_tree[ver][u]=span\_tree[u][ver];

edges++;

visit[ver]=1;

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

{

if(visit[i]==0&&cost[i][ver]<dist[i])

{

dist[i]=cost[i][ver];

via[i]=ver;

}

}

min\_cost=min\_cost+cost[u][ver];

}

cout<<"\nMinimum distance is::\t"<<min\_cost;

}

int main()

{

graph g;

g.input();

g.prims();

g.display();

return 0;

}

***Output :***

compeng-sl2-08@compeng-sl2-08:~/Priyanka$ g++ news.cpp

compeng-sl2-08@compeng-sl2-08:~/Priyanka$ ./a.out

Enter the number of vertices: 6

Enter the weights between1 and2: 1

Enter the weights between1 and3: 9

Enter the weights between1 and4: 3

Enter the weights between1 and5: 6

Enter the weights between1 and6: 9

Enter the weights between2 and3: 8

Enter the weights between2 and4: 2

Enter the weights between2 and5: 5

Enter the weights between2 and6: 4

Enter the weights between3 and4: 8

Enter the weights between3 and5: 2

Enter the weights between3 and6: 3

Enter the weights between4 and5: 10

Enter the weights between4 and6: 11

Enter the weights between5 and6: 7

Minimum distance is:: 12

The matrix representation is:

999 1 9 3 6 9

1 999 8 2 5 4

9 8 999 8 2 3

3 2 8 999 10 11

6 5 2 10 999 7

9 4 3 11 7 999

The spanning tree is:

0 1 0 0 0 0

1 0 0 2 0 4

0 0 0 0 2 3

0 2 0 0 0 0

0 0 2 0 0 0

0 4 3 0 0 0