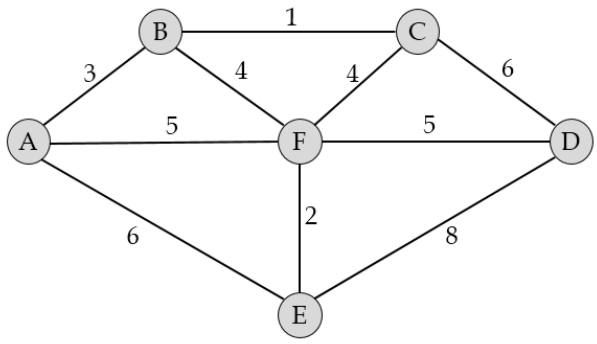
```
/*C Program to implement Kruskal's Algorithm
Input  : 1. No of vertices in the graph
         2. Cost adjacency matrix
Output : 1. Edges in the minimum spanning tree
         2. Weight of the minimum spanning tree
*/

#include<stdio.h>
#include<stdlib.h>

int i,j,k,a,b,u,v,n,ne=1;
int min,minCost=0,cost[9][9],parent[9];
int find(int);
int uni(int,int);

int main()
{
 printf("\n Enter the no. of vertices\n");
 scanf("%d",&n);
 printf("\n Enter the cost adjacency matrix\n");
 for(i=1;i<=n;i++)
 {
  for(j=1;j<=n;j++)
  {
   scanf("%d",&cost[i][j]);
   if(cost[i][j]==0)
    cost[i][j]=999;
  }
 }
 printf("\n");
 printf("Output:\n");
```

```
printf("\n");
printf("\n The edges of Minimum Spanning Tree are\n\n");

while(ne<n)
{
 for(i=1,min=999;i<=n;i++)
 {
  for(j=1;j<=n;j++)
  {
   if(cost[i][j]<min)
   {
    min=cost[i][j];
    a=u=i;
    b=v=j;
   }
  }
 }
 u=find(u);
 v=find(v);

 if(uni(u,v))
 {
  printf("\n%d edge (%d,%d) =%d\n",ne++,a,b,min);
  minCost +=min;
 }
 cost[a][b]=cost[b][a]=999;
}
printf("\n\tWeight of the minimum spanning tree = %d\n",minCost);

}
```

```
int find(int i)
{
 while(parent[i])
  i=parent[i];
 return i;
}

int uni(int i,int j)
{
 if(i!=j)
 {
  parent[j]=i;
  return 1;
 }
 return 0;
}
```

Sample Input and Output:




```
 Enter the cost adjacency matrix
 0   3   999   999   6   5
 3   0    1    999  999  4
999  1    0     6   999  4
999 999   6     0    8   5
 6  999  999    8    0   2
 5   4    4     5    2   0

Output:


 The edges of Minimum Spanning Tree are


1 edge (2,3) =1

2 edge (5,6) =2

3 edge (1,2) =3

4 edge (2,6) =4

5 edge (4,6) =5

        Weight of the minimum spanning tree = 15
Press any key to continue...
```