Experiment No. 8

Aim: Implement Minimum cost spanning trees-Kruskal algorithm

Theoretical Background:

It falls under a class of algorithms called greedy algorithms that find the local optimum in the hopes of finding a global optimum.
We start from the edges with the lowest weight and keep adding edges until we reach our goal.

The steps for implementing Kruskal's algorithm are as follows:

1. Sort all the edges from low weight to high
2. Take the edge with the lowest weight and add it to the spanning tree. If adding the edge created a cycle, then reject this edge.
3. Keep adding edges until we reach all vertices.

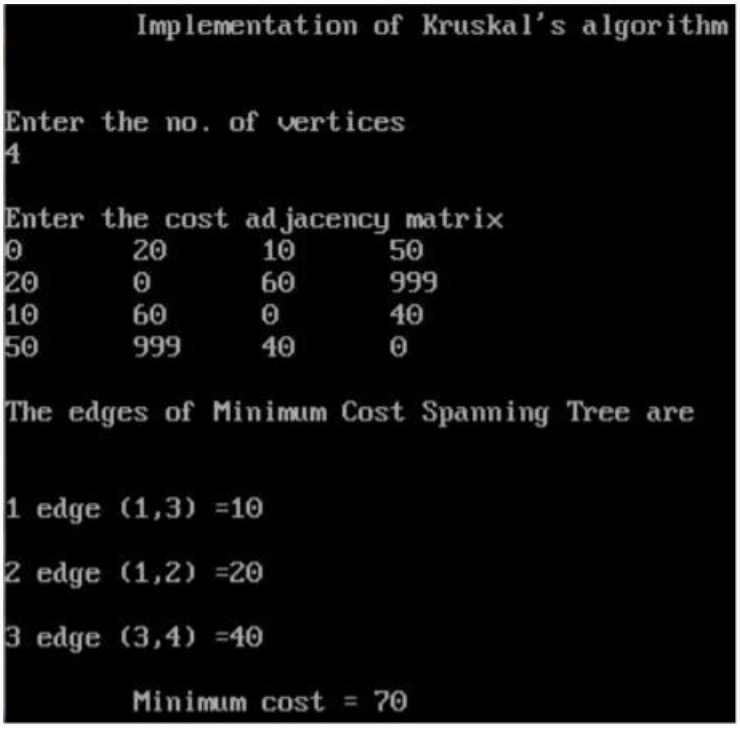
Program:

```
#include<stdio.h>
#include<conio.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);
void main()
{
clrscr();
printf("\n\n\tImplementation of Kruskal's algorithm\n\n");
printf("\nEnter the no. of vertices\n");
scanf("%d",&n);
```

```
printf("\nEnter 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("\nThe edges of Minimum Cost 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\tMinimum cost = %d\n",mincost);
getch();
}
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;
}
```

Output:



Conclusion:

Thus, in this experiment we have studied about implementing Minimum cost spanning trees using Kruskal algorithm