**EXPERIMENT – 9**

**AIM: Implement Kruskal’s algorithm**

**Input:**

#include<stdio.h>

#define INFINITY 999

typedef struct Graph

{

int v1;

int v2;

int cost;

}GR;

GR G[20];

int tot\_edges,tot\_nodes;

void create();

void spanning\_tree();

int Minimum(int);

void main()

{

printf("\n\t Graph Creation by adjacency matrix ");

create();

spanning\_tree();

}

void create()

{

int k;

printf("\n Enter Total number of nodes: ");

scanf("%d",&tot\_nodes);

printf("\n Enter Total number of edges: ");

scanf("%d",&tot\_edges);

for(k=0;k<tot\_edges;k++)

{

printf("\n Enter Edge in (V1 V2)form ");

scanf("%d%d",&G[k].v1,&G[k].v2);

printf("\n Enter Corresponding Cost ");

scanf("%d",&G[k].cost);

}

}

void spanning\_tree()

{

int count,k,v1,v2,i,j,tree[10][10],pos,parent[10];

int sum;

int Find(int v2,int parent[]);

void Union(int i,int j,int parent[]);

count=0;

k=0;

sum=0;

for(i=0;i<tot\_nodes;i++)

parent[i]=i;

while(count!=tot\_nodes-1)

{

pos=Minimum(tot\_edges);//finding the minimum cost edge

if(pos==-1)//Perhaps no node in the graph

break;

v1=G[pos].v1;

v2=G[pos].v2;

i=Find(v1,parent);

j=Find(v2,parent);

if(i!=j)

{

tree[k][0]=v1;//storing the minimum edge in array tree[]

tree[k][1]=v2;

k++;

count++;

sum+=G[pos].cost;//accumulating the total cost of MST

Union(i,j,parent);

}

G[pos].cost=INFINITY;

}

if(count==tot\_nodes-1)

{

printf("\n Spanning tree is...");

printf("\n--------------------------\n");

for(i=0;i<tot\_nodes-1;i++)

{

printf("[%d",tree[i][0]);

printf(" - ");

printf("%d",tree[i][1]);

printf("]");

}

printf("\n--------------------------");

printf("\nCost of Spanning Tree is = %d",sum);

}

else

{

printf("There is no Spanning Tree");

}

}

int Minimum(int n)

{

int i,small,pos;

small=INFINITY;

pos=-1;

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

{

if(G[i].cost<small)

{

small=G[i].cost;

pos=i;

}

}

return pos;

}

int Find(int v2,int parent[])

{

while(parent[v2]!=v2)

{

v2=parent[v2];

}

return v2;

}

void Union(int i,int j,int parent[])

{

if(i<j)

parent[j]=i;

else

parent[i]=j;

}