Graphs – Minimum Spanning Trees

By,

Vishwas Gajawada,

197188,

CSE –'A' section.

Code 1 – Prim's MST

```
#include<bits/stdc++.h>
using namespace std;

void input(int **&G,int v,int e,int x[],int y[],int w[]){
    G = new int*[v+1];
    for(int i=0;i<v+1;i++)G[i]=new int[v+1];
    for(int i=0;i<=v;i++)
        for(int j=0;j<=v;j++)
            G[i][j]=0;
    for(int i=0;i<e;i++){
        G[x[i]][y[i]]=w[i];G[y[i]][x[i]]=w[i];
    }
}

void prims_mst (int **G,int v){
    int visit[v+1]={0};
    int dist[v+1];int next[v+1]={0};
    for(int i=0;i<=v;i++)dist[i]=9999;
    int cur=1;
    visit[cur]=1;int ans=0,min;
    for(int j=0;j<6;j++){
        visit[cur]=1;int flag=0;
        for(int z=1;z<=v;z++){
            if(visit[z]==0)continue;
            dist[z]=9999;
            for(int i=1;i<=v;i++){
                if(G[z][i]>0 && visit[i]==0 && i!=z){
                    if(G[z][i]<dist[z]){
                        dist[z]=G[z][i];
                        next[z]=i;
                    }
                    flag=1;
                }
            }
            if(flag==0)dist[z]=-1;
        }
        min=9999;int temp=cur;
```

```
        for(int i=1;i<=v;i++){
            if(visit[i]==1 && dist[i]!=-1 && dist[i]<min){
                cur=i;min=dist[i];
            }
        }
        ans+=min;
        cout<<cur<<" "<<next[cur]<<endl;
        cur=next[cur];
    }
    cout<<ans;
}

int main()
{
    int **G;
    int v=7,e=12;
    int x[]={1,1,1,2,2,3,3,4,4,5,5,6};
    int y[]={2,3,4,4,5,4,6,6,7,4,7,7};
    int w[]={2,4,1,2,10,2,5,8,4,7,6,1};
    // // undirected
    input(G,v,e,x,y,w);
    prims_mst(G,v);
}
```

Output:

```
PS C:\Users\Vishwas Gajawada\Desktop\c++ codes\GRAPHS> cd "c
1 4
1 2
4 3
4 7
7 6
7 5
16
```

Code 2 – Krushkal's MST

```
#include<bits/stdc++.h>
using namespace std;

struct edge{
    int v1,v2,weight;
};
void swap(struct edge &a,struct edge &b){
    struct edge temp = a;
    a=b;b=temp;
}
struct minheap{//for maintaining min weight edge at top
    struct edge e[100];
    int s=0;
    void insert(int x,int y,int z){
        e[s].v1=x,e[s].v2=y,e[s].weight=z;
        int k=s;
        while(e[k].weight<e[(k-1)/2].weight){
            swap(e[k],e[(k-1)/2]);
            k = (k-1)/2;
        }
        s++;
    }
    struct edge pop()
    {
        struct edge ans=e[0];
        e[0] = e[--s];
        int i=0;
        int l=2*i+1,r=2*i+2;
        while((l<=s)||(r<=s)){
            int small=i;
            if(l<=s && e[small].weight>e[l].weight)small=l;
            if(r<=s && e[small].weight>e[r].weight)small=r;
            if(small!=i)swap(e[small],e[i]);
            if(small==i)break;
            i=small;
            l=2*i+1,r=2*i+2;
        }
        return ans;
    }
};
```

```
void input(int **&G,int v,int e,int x[],int y[],int w[],struct minhe
ap &m){
    G = new int*[v+1];
    for(int i=0;i<v+1;i++)G[i]=new int[v+1];
    for(int i=0;i<=v;i++)
        for(int j=0;j<=v;j++)
            G[i][j]=0;
    for(int i=0;i<e;i++){
        G[x[i]][y[i]]=w[i];G[y[i]][x[i]]=w[i];
        m.insert(x[i],y[i],w[i]);
    }
}
//disjoint set functions
int find(int s[],int v,int x){
    if(s[x]==-1)return x;
    else return find(s,v,s[x]);
}
void uni(int s[],int v,int x,int y){
    if(s[y]==-1)s[y]=x;
    else if(s[x]==-1)s[x]=y;
    else s[find(s,v,y)]=find(s,v,x);
}

int sum(int a[],int n){
    int s=0;
    for(int i=1;i<=n;i++)s+=a[i];
    return s;
}

//krushkal's mst
void krush_mst(int **G,int v,struct minheap m){
    int visit[v+1]={0};
    int s[v+1];for(int i=0;i<=v;i++)s[i]=-1;
    int ans=0;
    while(true){
        struct edge temp = m.pop();
        int x=temp.v1,y=temp.v2,z=temp.weight;
        if(find(s,v,x)!=find(s,v,y)){
            uni(s,v,x,y);
            ans+=z;
            visit[x]=visit[y]=1;
```

```
            cout<<x<<" "<<y<<endl;
        }
        if(sum(visit,v)==v)break;//all nodes covered
    }
    cout<<ans;
}

int main()
{
    int **G;
    int v=7,e=12;
    int x[]={1,1,1,2,2,3,3,4,4,5,5,6};
    int y[]={2,3,4,4,5,4,6,6,7,4,7,7};
    int w[]={2,4,1,3,10,2,5,8,4,7,6,1};
    struct minheap m;
    // // undirected
    input(G,v,e,x,y,w,m);
    krush_mst(G,v,m);

}
```

Output:

```
PS C:\Users\Vishwas Gajawada\Desktop\c++ codes\GRAPHS> cd "c:\Users\Vis
1 4
6 7
3 4
1 2
4 7
5 7
16
PS C:\Users\Vishwas Gajawada\Desktop\c++ codes\GRAPHS>
```