```cpp
/// Kruskal Algorithm(MST):
#include<bits/stdc++.h>
using namespace std;
#define mx 100
int par[mx];
struct edge{
    int u,v,w;
    edge(int a, int b, int c){
        u=a;v=b;w=c;
    }
    bool operator < (const edge& p) const{
        return p.w < w;
    }
};
priority_queue<edge>pq;
queue<edge>Q;

int findparent(int x){
    if(par[x]==x) return x;
    else return par[x]=findparent(par[x]);
}

int kruskalMST(int node){
    int Mincost=0;
    for(int i=1; i<=node; i++) par[i]=i;

    for(int i=1; i<=node-1; i++){
        label:
        edge top = pq.top();
        pq.pop();

        int parU = findparent(top.u);
        int parV = findparent(top.v);

        if(parU==parV){
            goto label;
        }else{
            par[parU]=parV;
            Q.push(top);
            Mincost += top.w;
        }
    }
    return Mincost;
}

int main(){
    int node,edg,u,v,w,mincst,src;
    cin >> node >> edg;

    for(int i=0; i<edg; i++){
        cin >> u >> v >> w;
        pq.push(edge(u,v,w));
    }

    mincst=kruskalMST(node);

    cout <<endl<<"Minimum Spanning Tree:"<<endl<<endl;
    cout << " Edges " << "  Weight" << endl;

    for(int i=1; i<=node-1; i++){
        edge top=Q.front();
        Q.pop();
        printf("%2d  %2d   %3d\n\n",top.u,top.v,top.w);
    }
    cout <<endl<<"Total Minimum Cost: "<<mincst<<endl;

    return 0;
}
```