Kruskal

点很少，按边权值排序，枚举枚举L和R，并查集检查连通性。一旦连通，那么更新答案。

最大边权与 最小边权差最小，肯定是找最小生成树，但却不一定是以所有边中边权最小的那个边为起点的边是满足题意的生成树，所以从小到大依次枚举每个边，让他们作为边权最小边，维护答案

#include<cstdio>

#include<algorithm>

#include<cstring>

#include<iostream>

using namespace std;

typedef long long ll;

const int INF=1000000007;

int n,m;

struct node

{

int u,v,w;

}edge[10010];

int fa[110];

bool cmp(node a,node b)

{

return a.w<b.w;

}

int Find(int x)

{

if(fa[x]==x)return x;

return fa[x]=Find(fa[x]);//是否需要带路径压缩还要考虑一下,有时带了会超时

//return Find(f[x]); //不路径压缩版

}

inline void init()

{

for(int i=1;i<=n;i++)

fa[i]=i;

}

int main()

{

//freopen("input.txt","r",stdin);

while(cin>>n>>m && (n+m))

{

for(int i=0;i<m;i++)

cin>>edge[i].u>>edge[i].v>>edge[i].w;

sort(edge,edge+m,cmp);

int ans(INF);

for(int i=0;i<m;i++)

{

init();

int cnt=n-1;

int maxl=0;

for(int j=i;j<m;j++)

{

int du=Find(edge[j].u);

int dv=Find(edge[j].v);

if(du!=dv)

{

fa[dv]=du;

maxl=edge[j].w;

cnt--;

if(cnt==0)

{

ans=min(ans,maxl-edge[i].w);

break;

}

}

}

}

if(ans==INF)ans=-1;

cout<<ans<<endl;

}

return 0;

}