如果是满足题中条件的话，那么一个集合里的所有元素的边数都应该相等，等于集合的元素个数-1（-1是自己）

#include <bits/stdc++.h>

using namespace std;

int fa[150010];

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

}

void Merge(int x,int y)//不按秩合并

{

x=Find(x);

y=Find(y);

if(x!=y)

fa[y]=x;

}

int degree[150010];

int main()

{

int n,m;

scanf("%d%d",&n,&m);

memset(fa,0,sizeof(fa));

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

fa[i]=i;

int x,y;

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

{

scanf("%d%d",&x,&y);

Merge (x,y);

degree[x]++;

degree[y]++;

}

map<int,int>\_map;

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

\_map[Find(i)]++;

int flag=1;

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

{

if(\_map[Find(i)]-1!=degree[i])

{

flag=0;

break;

}

}

if(flag)

puts("YES");

else puts("NO");

return 0;

}

法2：

也就是每个联通块的点数n和边数m要满足关系m=n\*（n-1）/2

每个集合中的任两点都是有边相连的，也就是说一个集合中的边数=点数n\*(n-1)/2

那么所有集合的边数相加应该=m

#include <bits/stdc++.h>

using namespace std;

#define ll long long

int fa[150010];

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

}

void Merge(int x,int y)//不按秩合并

{

x=Find(x);

y=Find(y);

if(x!=y)

fa[y]=x;

}

int main()

{

int n,m;

scanf("%d%d",&n,&m);

memset(fa,0,sizeof(fa));

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

fa[i]=i;

int x,y;

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

{

scanf("%d%d",&x,&y);

Merge (x,y);

}

map<int,int>\_map;

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

\_map[Find(i)]++;

ll sum=0;

for(auto it:\_map)

{

sum+=(ll)it.second\*(it.second-1)/2ll;

}

if(sum==m)

puts("YES");

else puts("NO");

return 0;

}

法3：DFS

也可以用dfs找联通块