[并查集2个优化——按秩合并和路径压缩](https://www.cnblogs.com/oiercc/p/7210136.html)

并查集有两个优化。

一、按秩合并

描述：就是在对两个不同子集连接时，按照rank来连，也就是rank低的连在rank高的下面。rank高的做父亲节点。

作用，这样类似维护了一棵树，树是rank高的在上。

下面代码供参考，另外在外面有相关代码

void init(int n)  // 初始化n个元素

{

    for(int i=0;i<n;i++)

    {

        parent[i]=i;

        rank[i]=0;   // 初始树的高度为0

    }

}

// 合并x和y所属的集合

void unite(int x,int y)

{

    x=find(x);

    y=find(y);

    if(x==y) return ;

    if(rank[x]<rank[y])

        parent[x]=y;  // 合并是从rank小的向rank大的连边

    else

    {

        parent[y]=x;

        if(rank[x]==rank[y]) rank[x]++;

    }

}

二、路径压缩

描述：假如fa数组已经嵌套了N层，那么传统的做法去找祖先要做N次，当N很大时，这种做法很没效率。

比较常见的是采用递归路径压缩的方法查找元素（见相关代码），但是，递归压缩路径可能会造成溢出栈，会发生RE。

下面我们说一下非递归方式进行的路径压缩：

int find(int x)

{

int k, j, r;

r = x;

while(r != parent[r]) //查找跟节点

r = parent[r]; //找到跟节点，用r记录下

k = x;

while(k != r) //非递归路径压缩操作

{

j = parent[k]; //用j暂存parent[k]的父节点

parent[k] = r; //parent[x]指向跟节点

k = j; //k移到父节点

}

return r; //返回根节点的值

}