Boruvka算法代码实现

主讲人：邓哲也

边使用数组存储即可

```
struct edge {
    int u, v, w;
}e[10010];
```

并查集的操作

```
int FindSet(int x) {
    if (x != parent[x])
        parent[x] = FindSet(parent[x]);
    return parent[x];
}
```

并查集的操作

```
int Union(int x, int y) {
        x = FindSet(x);
        y = FindSet(y);
        if (x == y) return 0;
        if (rk[x] > rk[y])  parent[y] = x;
        else {
                parent[x] = y;
                if (rk[x] == rk[y])
                        rk[y] ++;
        }
        return 1;
}
```

Boruvka主体部分（一）

```
while (1) {
    int isEnd = 1;
    for (int i = 1;i <= n;i ++)
        if (FindSet(i) != FindSet(1)) {
            isEnd = 0;
            break;
        }
    if (isEnd) break;
```

Boruvka主体部分（二）

```
for (int i = 1;i <= n;i ++) {
        low[i] = 0x3f3f3f3f, pos[i] = 0;
        parent[i] = FindSet(parent[i]);
}
for (int i = 1;i <= m;i ++) {
        int u = FindSet(e[i].u);
        int v = FindSet(e[i].v);
        if (u != v){
                if (e[i].w < low[u]) low[u] = e[i].w, pos[u] = i;
                if (e[i].w < low[v]) low[v] = e[i].w, pos[v] = i;
        }
}
```

Boruvka主体部分（三）

```
for (int i = 1;i <= n;i ++)
    if (pos[i] > 0 && Union(e[pos[i]].u, e[pos[i]].v))
        ans += e[pos[i]].w;
}
```

试一试吧！—— HDU 1233

输入格式

每个测试用例的第1行给出村庄数目N（< 100）；

随后的N(N-1)/2行对应村庄间的距离，每行给出一对正整数，分别是两个村庄的编号，以及此两村庄间的距离。

为简单起见，村庄从1到N编号。

输出格式

对每个测试用例，在1行里输出最小的公路总长度。

下节课再见