```
#include <bits/stdc++.h>
using namespace std;
class DisjointSet {
   vector<int> rank, parent, size;
public:
   DisjointSet(int n) {
      rank.resize(n + 1, 0);
      parent.resize(n + 1);
      size.resize(n + 1);
      for (int i = 0; i <= n; i++) {
         parent[i] = i;
         size[i] = 1;
      }
   }

   int findUPar(int node) {
      if (node == parent[node])
         return node;
      return parent[node] = findUPar(parent[node]);
   }

   void unionByRank(int u, int v) {
      int ulp_u = findUPar(u);
      int ulp_v = findUPar(v);
      if (ulp_u == ulp_v) return;
      if (rank[ulp_u] < rank[ulp_v]) {
         parent[ulp_u] = ulp_v;
      }
      else if (rank[ulp_v] < rank[ulp_u]) {
         parent[ulp_v] = ulp_u;
      }
```

```
        else {
            parent[ulp_v] = ulp_u;
            rank[ulp_u]++;
        }
    }

    void unionBySize(int u, int v) {
        int ulp_u = findUPar(u);
        int ulp_v = findUPar(v);
        if (ulp_u == ulp_v) return;
        if (size[ulp_u] < size[ulp_v]) {
            parent[ulp_u] = ulp_v;
            size[ulp_v] += size[ulp_u];
        }
        else {
            parent[ulp_v] = ulp_u;
            size[ulp_u] += size[ulp_v];
        }
    }
};
```