HONG – LOU 2022 – 8 – 23

当时第一次看到巨佬写的一行$find$函数就觉得这个东西好帅，后来在CF也看到很多DSU的题目，现在写篇笔记系统学习一下。

orzorz

弔背景

话说江湖上散落着各式各样的大侠，有上千个之多。他们没有什么正当职业，整天背着剑在外面走来走去，碰到和自己不是一路人的，就免不了要打一架。但大侠们有一个优点就是讲义气，绝对不打自己的朋友。而且他们信奉“朋友的朋友就是我的朋友”，只要是能通过朋友关系串联起来的，不管拐了多少个弯，都认为是自己人。这样一来，江湖上就形成了一个一个的帮派，通过两两之间的朋友关系串联起来。而不在同一个帮派的人，无论如何都无法通过朋友关系连起来，于是就可以放心往死了打。

初始化：

在江湖上，有非常多的英雄，我们不妨用一个f数组来保存每位英雄的掌门。

```
const int X = 10010;
int f[X];
```

在帮派中，有掌门和弟子，那么刚刚开始肯定都是一个人行走江湖，所以在程序初始化的时候，每个人的掌门都是他们自己。

```
void init() {
        for(int i = 1;i <= X;i++){
                f[i] = i;
        }
}
```

查找根节点

我们在判断两位英雄是否师出同门的时候，要用到查找掌门的函数。

这里我们用了记忆化，俗称“压缩路径”。

```
int find(int x) {
  if(x != f[x]){
    return f[x] = find(f[x]); //在递归的时候，就直接将遇到的当前帮派的英雄的掌门修改了
  }
  return f[x]; //如果找到了掌门，就直接返回掌门编号
}
```

一种更帅的写法

```
int find(int x) {
  return f[x] == x ? x : f[x] = find(f[x]);
}
```

合并子集

在确认两位英雄是属于同一个帮派的时候，要把两位英雄的帮派合并，既然师出同门，那两个帮派就是一样的了嘛。

```
void join(int x,int y) {
  int fx = find(x), fy = find(y); //找到两位英雄的掌门
  if(fx != fy){
    f[fy] = fx; //合并子集
  }
}
```

同样，一种更帅的写法

```
void merge(int x, int y) {
  f[find(x)] = find(y);
}
```

主函数

接下来就是整个完整的程序。

```cpp
/**
 *    author:  HONG-LOU
 *    created: 2022-08-23 10:43:27
**/
#include <bits/stdc++.h>

int f[202020];

int find(int x) {return x == f[x] ? x : f[x] = find(f[x]);}

int main() {
  std::ios::sync_with_stdio(false);
  std::cin.tie(nullptr);

  int n; std::cin >> n;

  for(int i = 1; i <= n; i++) f[i] = i;

  int m; std::cin >> m;

  for(int i = 0; i < m; i++) {
    int x, y, z; std::cin >> x >> y >> z;

    if(x == 1) f[find(y)] = find(z);
    else std::cout << (find(y) == find(z) ? "Y\n" : "N\n");
  }
  return 0;
}
```

完结撒花！！！