食物链

题目链接

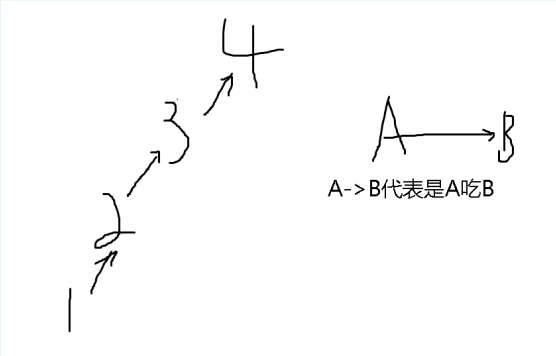
acwing.240

食物链是并查集里面的一个题目涉及**路径压缩**和**距离记录**

核心代码

```
int find(int x)
{
    if(p[x]!=x)
    {
        int u = find(p[x]);//找到父节点的父节点，那么最终都是找到根节点的位置
    d[x]+=d[p[x]];//d[x]是本身x到其父节点的位置，让x到根节点的位置进行压缩
    p[x]=u;
     }
return p[x];
}
```

其实在这其中压缩是比较难以理解的



假设有这个一条链 存在者 1吃 2，2吃3，3吃4

那么可以确定在加入的时候的距离最开始为

1和2距离为1，2和3距离为1,3和4距离为1

因为在压缩过程中**最终的父节点**为根节点，那么 d[3] = 1； d[2] = 1(此时是到它父节点也就是3的距离) d[2]+=d[3] 这就是在压缩过程，那么最终2的父节点会变为4，d[2]=2 ， 同理 d[1] = 3；

因为有相同的根节点才能根据**相距根节点**来判断两者什么关系

有了这层关系后那么就可以根据以上信息写出代码

代码如下

```
#include<iostream>

using namespace std;
const int N = 1e5+10;
int p[N],d[N];//di是i距离父节点的位置
int n,m;
int main()
```

```
{
cin >> n >> m;
for(int i = 1; i <= n; i ++) p[i] = i;//开始让他们自己的父节点为自己
int res = 0;
while(m--)
{
   int a,b,t;
   cin >> t >> a >> b;
   if(a>n||b>n) res++;
   else
   {
       int px = find(x),py = find(y);//先找到父节点的位置后面要用
       if(t==1)// 即同类情况
       {
           if(px==py&&(d[x]-d[y])%3) //如果在一条链上，同时(dx-dy)%3不为0那么两者不是同类 比如有一个1->2->3->4->5->6->7
   // x  y  z x  y  z  x假设上面的食物链是这种关系 x吃y y吃z z吃x 那么判断 1和7 的时候就需要距离来判断了  d[1] = 6, d[7] = 0, 那么(d[1]-d[7])%3==0所以他们为同类
           {
               res++;
           }else if(px!=py)//不在同一链上那么就加入
           {
               p[px] = py;
               d[px] = d[y] - d[x];//这个就是让他俩加入比如 1  2->3->4，1和3不在同一链上，那么让1与3的父节点相同 再让 1父节点到它父节点的父节点(其实是他本身1)的距离变为两者之间的距离d[1] = 0,d[3] = 1,那么1父节点为4  同时距离父节点的位置为1 就可以 d[1] = 1(更新后)那么更新后变为了 2->3->4和2->1->4
           }
       }else//t == 2 的时候基本同理
       {
           if(px==py&&(d[x]-d[y]-1)%3) res++;
           else if(px!=py)
           {
               p[px] = py;
               d[px] = d[y] - d[x] + 1;
           }
       }
   }
}
cout << res;
}

```