K题题解

题目

有一个遥远的国度，居民们都住在房屋里。房屋之间任意可达，且任意两个房屋之间有且只有一条无向路径。这个国度的人充满好奇心，当两个房屋里的人见面之前，双方都想要知道他们见面所需行走的最短路径长度之和。

Standard Input

第一行两个正整数 n*n* 和 m*m* (2\leq n\leq 5\times10^5,2\leq m\leq 5\times10^5)
(2≤*n*≤5×105,2≤*m*≤5×105)，分别表示共 n*n* 个房屋，有 m*m* 次询问。
接下来 n-1*n*−1 行，每行两个整数 u_i*u***i* 和 v_i*v***i* (1\leq u_i\leq n,1\leq v_i\leq n)(1≤*u***i*≤*n*,1≤*v***i*≤*n*)，表示编号为 u*u* 和 v*v* 的房屋被一条长为 11 的路径相连。
接下来 m*m* 行，每行两个整数 a_i*a***i* 和 b_i*b***i* (1\leq a_i\leq n,1\leq b_i\leq n)(1≤*a***i*≤*n*,1≤*b***i*≤*n*)，表示询问编号为 a*a* 和 b*b* 的房屋间的最短距离。
输入数据保证合法。

Standard Output

输出共 m*m* 行，每行有且只有一个正整数，是对于第 i*i* 次询问的回答，即两个房屋间的最短距离。

Samples

Input	Output
`6 3 1 2 1 5 5 3 5 6 3 4 1 6 2 4 3 5`	`2 4 1`

Problem ID	2723
Problem Title	居民们都住在房屋里
Time Limit	1000 ms
Memory Limit	64 MiB
Output Limit	64 MiB
Source	2022 UESTC ICPC Training for Graph

题解

根据题意可知，由于只有n-1条边，所以这个问题发生在树上，我们在树上求两点之间的最短距离：

不妨设每个节点的深度数组：$dep[x]$表示x的深度。

则点i和点j的最短距离：$MinDistance = dep[i] + dep[j] - 2 * dep[lca(i,j)]$

所以问题的重点转化为求点i,j的$LCA(i,j)$

关于最近公共祖先LCA

求最近公共祖先可以使用想上标记或者倍增的方法，这里重点介绍倍增法。

所谓倍增，就是按 2 的倍数来增大，也就是跳 $1, 2, 4, 8, 16, 32 \ldots$ 不过在这我们不是按从小到大跳，而是从大向小跳，即按$\ldots\ldots 32, 16, 8, 4, 2, 1$ 来跳，如果大的跳不过去，再把它调小。这是因为从小开始跳，可能会出现 “悔棋” 的现象。拿 5 为例，从小向大跳，$5 \neq 1 + 2 + 4$, 所以我们还要回溯一步，然后才能得出$5 = 1 + 4$；而从大向小跳，直接可以得出$5 = 4 + 1$。这也可以拿二进制为例，$5(101)$，从高位向低位填很简单，如果填了这位之后比原数大了，那我就不填，这个过程是很好操作的。

我们先用dfs求出来fa数组

$fa[x][y]$表示x向上数2^y位是谁。

然后先把xy调整到一致的高度，再一起往上翻，就可以找到我们要找的LCA

代码

```
#include <bits/stdc++.h>
using namespace std;
int n,m;
const int MAXN=1000005;
int head[MAXN],ver[MAXN],nxt[MAXN],tot;
int qx[MAXN],qy[MAXN],cnt;
int dis[MAXN];
int fa[MAXN][30],deep[MAXN];
bool vis[MAXN];
int pw[30];
inline void add(int x,int y)
{
    ver[++tot]=y;nxt[tot]=head[x];head[x]=tot;
}
void dfs(int x,int f)
{
    vis[x]=1;
    fa[x][0]=f;
    deep[x]=deep[f]+1;
    for(int i=1;pw[i]<=deep[x];i++)
        fa[x][i]=fa[fa[x][i-1]][i-1];
    for(int i=head[x];i;i=nxt[i])
    {
        int y=ver[i];
        if(vis[y]==1)continue;
        if(y==f)continue;
        dfs(y,x);
    }
    vis[x]=0;
}
inline int LCA(int x,int y)
{
    if(deep[x]>deep[y])swap(x,y);
    for(int i=25;i>=0;i--)
    {
        if(!fa[y][i])continue;
        if(deep[fa[y][i]]>=deep[x])
        {
```

```
            y=fa[y][i];
        }
    }
    // cout<<" x  y  "<<x<<" "<<y<<endl;
    if(x==y)return x;
    for(int i=25;i>=0;i--)
    {
        if(!fa[x][i])continue;
        if(!fa[y][i])continue;
        if(fa[x][i]!=fa[y][i])
        {
            x=fa[x][i];
            y=fa[y][i];
        }
    }
    return fa[x][0];
}
int main()
{
    scanf("%d %d",&n,&m);
    pw[0]=1;
    for(int i=1;i<=29;i++)
        pw[i]=pw[i-1]<<1;
    for(int i=1;i<n;i++)
    {
        int x,y;
        scanf("%d %d",&x,&y);
        add(x,y);add(y,x);
    }
    deep[1]=1;
    dfs(1,1);
    for(int i=1;i<=m;i++)
    {
        int x,y;
        scanf("%d %d",&x,&y);
        int lca=LCA(x,y);
        // cout<<"lca "<<lca<<endl;
        printf("%d\n",deep[x]-deep[lca]+deep[y]-deep[lca]);
    }
    return 0;
}
```