#include<iostream>

#include<queue>

#include<stack>

using namespace std;

const int N=1e5+10;

struct Edge{//边的结构体

int to,nxt;

}e[N\*2];

int adt,head[N];

void add(int u,int v){//加边建树

e[++adt]={v,head[u]};

head[u]=adt;

}

int fa[N],cntp[N];

void dfs(int p1){//递归实现dfs

cntp[p1]=0;

for(int i=head[p1];i!=0;i=e[i].nxt){

int p2=e[i].to;

if(p2==fa[p1]) continue;

fa[p2]=p1;//点p2的父节点是点p1

dfs(p2);

cntp[p1]++;//统计子节点个数

}

}

void dfs2(){//栈实现dfs

stack<int> s;

s.push(1);

while(!s.empty()){

int p1=s.top();//访问栈顶

s.pop();//出栈

cntp[p1]=0;

for(int i=head[p1];i!=0;i=e[i].nxt){

int p2=e[i].to;

if(p2==fa[p1]) continue;

fa[p2]=p1;//点p2的父节点是点p1

s.push(p2);//入栈

cntp[p1]++;//统计子节点个数

}

}

}

void bfs(){//队列实现bfs

queue<int> q;

q.push(1);

while(!q.empty()){

int p1=q.front();//访问队首

q.pop();//出队

cntp[p1]=0;

for(int i=head[p1];i!=0;i=e[i].nxt){

int p2=e[i].to;

if(p2==fa[p1]) continue;

fa[p2]=p1;//点p2的父节点是点p1

q.push(p2);//入队

cntp[p1]++;//统计子节点个数

}

}

}

int main() {

int n;

cin>>n;

for(int i=1;i<=n-1;i++){

int u,v;

scanf("%d%d",&u,&v);

add(u,v);

add(v,u);

}

//dfs(1);

//bfs();

dfs2();

for(int i=1;i<=n;i++){

cout<<i<<": "<<cntp[i]<<endl;

}

return 0;

}

/\*

stdin

9

1 2

1 9

2 3

2 4

4 5

4 6

4 7

5 8

\*/

/\*

stdout

1: 2

2: 2

3: 0

4: 3

5: 1

6: 0

7: 0

8: 0

9: 0

\*/