//diameter of a tree

#include<bits/stdc++.h>

using namespace std;

vector<vector<int>>g;

int parent[100006];// an array to keep a track of parent of each tree

int dist[100008];

void edge(int u, int v)

{

g[u].push\_back(v);

g[v].push\_back(u);

}

void dfs(int x)

{

for(int i=0;i<g[x].size();i++)

if(g[x][i]!=parent[x])

{parent[g[x][i]] =x;

dist[g[x][i]]=dist[x]+1;

dfs(g[x][i]);

}

}

int main()

{

int n;

cin>>n;

for(int i=0;i<=n;i++)

{

parent[i]=-1;//initially we dont know anyones parents

}

g.assign(n+1, vector<int>());

for(int i=0;i<n-1;i++)

{

int u,v;

cin>>u>>v;

edge(u,v);

}

dist[1]=0;

dfs(1);

/\*for(int i=1;i<=n;i++)

{

cout<<dist[i];

}

cout<<"\n";\*/

int maxdist1,maxdist2;

int maxd=\*max\_element(dist+1,dist+n+1);

//cout<<maxd;

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

{

if(dist[i]==maxd)

{

maxdist1=i;//maxdist1 is the node with maxdist from one

break;

}

}

// cout<<maxdist1;

for(int i=0;i<=n;i++)

{

parent[i]=-1;//since we ae changing our root we need to re-initialize parents

}

dist[maxdist1]=0;

dfs(maxdist1);

/\* for(int i=1;i<=n;i++)

{

cout<<dist[i];

}

cout<<"\n";\*/

maxd=\*max\_element(dist+1,dist+n+1);

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

{

if(dist[i]==maxd)

{

maxdist2=i;//maxdist2 is the node with maxdist from maxdist1

break;

}

}

cout<<dist[maxdist2]+1;

return 0;

}