**Tree Dp: Codeforces 161D; Distance in a Tree**

A tree is a connected graph that doesn't contain any cycles.
The distance between two vertices of a tree is the length (in edges) of the shortest path between these vertices.
You are given a tree with n vertices and a positive number k. Find the number of distinct pairs of the vertices which have a distance of exactly k between them. Note that pairs (v, u) and (u, v) are considered to be the same pair.

The first line contains two integers n and k ($1 \le n \le 50000$, $1 \le k \le 500$) — the number of vertices and the required distance between the vertices.
Next n - 1 lines describe the edges as "$a_i$ $b_i$" (without the quotes) ($1 \le a_i, b_i \le n$, $a_i \ne b_i$), where ai and bi are the vertices connected by the i-th edge. All given edges are different.

Print a single integer — the number of distinct pairs of the tree's vertices which have a distance of exactly k between them.

**Sample Code:**

```
#include<bits/stdc++.h>
using namespace std;
int n,k;
int dp[50002][502], wp[50002][502];
vector<int>ed[50002];
void DFS(int u,int pr)
{
    int sz = ed[u].size();
    if(u!=1)sz--;
    if(sz==0){
        dp[u][0] = 1;
        return;
    }
    dp[u][0]=1;
    for(int i=0; i<ed[u].size(); i++)
    {
        int v = ed[u][i];
        if(v==pr)continue;

        DFS(v,u);

        for(int j=0; j<k; j++){
            wp[u][k] += dp[u][j]*dp[v][k-j-1];
        }

        for(int j=1; j<=k; j++){
```

```
            dp[u][j] += dp[v][j-1];
        }
    }
}
int main()
{
    scanf("%d%d",&n,&k);
    for(int i=1; i<n; i++){
        int u,v; scanf("%d%d",&u,&v);
        ed[u].push_back(v);
        ed[v].push_back(u);
    }

    DFS(1,0);

    int ans=0;
    for(int i=1; i<=n; i++){
        ans += wp[i][k];
    }
    printf("%d\n",ans);

    return 0;
}
```