题目

给定一棵树，可以选择若干组没有公共端点的边，同时还可以往里面加k组权值为p的边，求选择的边的最大权重。

分析

考虑每个点是否可以向上连边，以及每个点的子树删除恰好m个点的最大贡献

于是题目就是求数上背包

```
#include <cinttypes>
#include <map>
#include <set>
#include <array>
#include <cmath>
#include <queue>
#include <stack>
#include <tuple>
#include <bitset>
#include <cctype>
#include <cstdio>
#include <random>
#include <string>
#include <vector>
#include <cassert>
#include <cstring>
#include <iomanip>
#include <limits.h>
#include <iostream>
#include <algorithm>
#include <functional>
#include <unordered_map>
using namespace std;

#define N maxn
#define db double
#define il inline
#define fir first
#define sec second
#define eps (1e-8)
#define pb push_back
#define ll long long
#define mkp make_pair
#define eb emplace_back
#define pii pair<int, int>
#define lowbit(a) (a & (-a))
#define SZ(a) ((int)a.size())
#define ull unsigned long long
#define all(a) a.begin(), a.end()
#define split cout << "==========\n";
#define GG { cout << "NO\n"; return; }
#define pll pair<long long, long long>
#define equals(a, b) (fabs((a) - (b)) < eps)

constexpr int ON = 0;
```

```cpp
constexpr int CW = -1;
constexpr int CCW = 1;
constexpr int BACK = 2;
constexpr int FRONT = -2;
const db pi = acos(-1.000);
constexpr int maxn = 1e5 + 50;
constexpr int INF = 0x3f3f3f3f;
constexpr ll LINF =  0x3f3f3f3f3f3f3f3f;
constexpr int mod = 998244353; //1e9 + 7; /* 1e9 + 7 */
constexpr int dir[8][2] = {-1, 0, -1, 1, 0, 1, 1, 1, 1, 0, 1, -1, 0, -1, -1,
-1};

mt19937_64 rnd(random_device {}());
uniform_int_distribution<ull> dist(0, ULLONG_MAX);//use dist(rnd)

bool BEGIN;

vector<pair<int,ll>>G[maxn];
ll dp[maxn][210][2];
ll tmp[210][2];
int sz[maxn];
int n, k;
ll p;
void dfs(int x, int fa)
{
    for(int i = 0; i <= 2 * k; ++i) dp[x][i][0] = dp[x][i][1] = -LINF;
    dp[x][0][1] = 0;
    dp[x][1][0] = 0;
    sz[x] = 1;
    for(auto [y, w]:G[x])
    {
        if(y == fa) continue;
        dfs(y, x);
        for(int i = 0; i <= min(2 * k, sz[x] + sz[y]); ++i)
        {
            tmp[i][0] = -LINF;
            tmp[i][1] = -LINF;
        }
        for(int i = 0; i <= min(2 * k, sz[x]); ++i)
        {
            for(int j = 0; j <= min(2 * k, sz[y]) && i + j <= min(2 * k, sz[x] +
sz[y]); ++j)
            {
                tmp[i + j][1] = max({tmp[i + j][1], dp[x][i][1] + dp[y][j][0],
dp[x][i][1] + dp[y][j][1]});
                tmp[i + j][0] = max({tmp[i + j][0], dp[x][i][1] + dp[y][j][1] +
w, dp[x][i][0] + dp[y][j][1], dp[x][i][0] + dp[y][j][0]});
                if(i + j < 2 * k)
                {
                    tmp[i + j + 1][0] = max({tmp[i + j + 1][0], dp[x][i][1] +
dp[y][j][1], dp[x][i][1] + dp[y][j][0]});
                }

            }
        }
        sz[x] += sz[y];
        for(int i = 0; i <= min(2 * k, sz[x]); ++i)
        {
```

```
            dp[x][i][0] = tmp[i][0];
            dp[x][i][1] = tmp[i][1];
        }
    }

}
void solve() {

    cin >> n >> k >> p;
    for(int i = 1; i < n; ++i)
    {
        int u, v;
        ll w;
        cin >> u >> v >> w;
        G[u].emplace_back(v, w);
        G[v].emplace_back(u, w);
    }
    dfs(1, 0);
    ll ans = 0;
    for(int i = 0; i <= 2 * k; ++i)
    {
        ans = max(ans, (i / 2ll) * p + max(dp[1][i][0], dp[1][i][1]));
    }
    cout << ans << '\n';
}
bool END;
signed main() {
    // cout << fixed << setprecision(10);
    ios::sync_with_stdio(false); cin.tie(nullptr);
//     int T; cin >> T; while (T--)
//  for(n = 1; n <= 10; ++n)
    solve();
    // cout << ((&END - & BEGIN) >> 21) << '\n';
    return 0;
}
```