Can'ın Alt Ağacı Çözüm

3 uzunluğundaki bir alt ağacı ele aldığımızda iki tane leaf düğüm arasında ortanca kenarın bir tane olacağı sabittir. Yani başka bir deyişle öyle bir a,b düğümler bu alt ağaca dahildir ki, kalan butün düğümler bu a ile b düğümüne komşudur, çünkü a ile komşularının uzunluğu 1 birim, a ile b nin uzunluğu 1 birim, b ile komşularının 1 birim uzaklık olmak üzere toplamda 3 birim uzunluğunda bir alt ağaç oluşur.

O zaman çözüm için yapmamız gereken bütün kenarları deneyerek, bütün a-b ikililerini denemek.

Bunu yaparken her bir denememizi Log N zamanda yapmamız lazım ki toplam karmaşıklığımız O(N Log N) olsun. Bunu sağlamak için her düğüm için komşularını sıralı olarak tutuyoruz, ayrıca bu sırada prefix toplam dizisi tutuyoruz.

Şimdi her a-b ikilisi geldiğinde elimizde iki tane dizi var ve bu iki diziden toplamı en yüksek yapacak k-2 elemanı seçmemiz gerekiyor, bunu ikili arama(binary search) yaparak her diziden kaçar tane almamız gerektiğini bulabiliriz, çünkü ilk diziden en düşük x değerli elemanı alırsak ikinci diziden de x değerine kadar hepsini almamız gerek(eğer almasaydık x değeri yerine almadığımız değeri almak daha optimaldi). Bu sayede K-2 eleman aldıracak elamanları x değerine binary search uygulayarak bulabiliriz. Bu elemanların toplamını da baştan sonra sırayla toplarsak yavaş olacağı için önceden her düğüm için hesapladığımız prefix toplam dizisinden bulacağız.

Bu sayede toplamda N-1 kenarı gezip her kenar için Log N işlemde o kenarın alt ağacın parçası olduğu ihtimali bulmuş oluyoruz.

Çözüm Kodu:

```
#include <bits/stdc++.h>
#define st first
#define nd second
#define mp make_pair
#define pb push_back
#define orta ((bas+son)/2)
#define coc g[node][i]
#define mod 1000000007
#define inf 1000000009
#define N 200005
using namespace std;

typedef long long ll;
typedef pair < ll , ll > ii;

ll n, k, l, ans = -inf, a[N], u[N], v[N];
vector < pair < ll , ii > > g[N];
vector < ll > pre[N];
```

```
map < ll , ll > yer[N];

int main() {
	scanf("%lld %lld %lld",&n ,&l ,&k);
	for(ll i = 1; i <= n; i++)
		scanf("%lld", &a[i]);
	for(ll i = 1; i < n; i++){
		scanf("%lld %lld",&u[i] ,&v[i]);
		ll vyer = g[u[i]].size();
		ll uyer = g[v[i]].size();
		g[u[i]].pb(mp(a[v[i]], mp(v[i], uyer)));
		g[v[i]].pb(mp(a[u[i]], mp(u[i], vyer)));
	}
	for(ll i = 1; i <= n; i++){
		sort(g[i].begin(), g[i].end());
		reverse(g[i].begin(), g[i].end());
	}

	for(ll i = 1; i <= n; i++)
		for(ll j = 0; j < g[i].size(); j++)
			yer[g[i][j].nd.st][i] = j;

	for(ll i = 1; i <= n; i++)
		ans = max(ans, a[i]);
	if(k == 1){
		printf("%lld\n", ans);
		return 0;
	}

	for(ll i = 1; i <= n; i++){
		ll mx = 0, top = 0;
		for(ll j = 0; j < g[i].size(); j++){
			top += g[i][j].st;
			mx = max(mx, top);
			pre[i].pb(mx);
		}
	}
	for(ll i = 1; i <= n; i++){
		ll top = a[i];
		ans = max(ans, top);
		for(ll j = 0; j < min(k - 1, (ll)g[i].size()); j++){
			top += g[i][j].st;
			ans = max(ans, top);
		}
		ans = max(ans, top);
	}
	if(l==3){
		for(ll i = 1; i < n; i++){
```

```
                ll top = a[u[i]] + a[v[i]];
                ans = max(ans, top);
                ll geldi = 0;
                if((ll)g[u[i]].size() < (ll)g[v[i]].size()){
                     ll uyer = yer[u[i]][v[i]];
                     for(ll j = 0; j < min(k - 2, (ll)g[u[i]].size());
j++){
                          if(g[u[i]][j].nd.st == v[i]){
                               geldi = 1;
                               continue;
                          }
                          ll bak = k - (j - geldi + 1 + 2) - 1;
                          ll ek = 0;
                          if(bak >= 0){
                               if(bak >= uyer){
                                    if(a[u[i]] < 0){
                                         if(uyer >= 1)
                                              ek = pre[v[i]][uyer -
1];
                                    } else{
                                         ek = pre[v[i]][min(bak + 1,
(ll)pre[v[i]].size() - 1)] - a[u[i]];
                                    }
                               }else
                                    ek      =      pre[v[i]][min(bak,
(ll)pre[v[i]].size() - 1)];
                          }
                          top += g[u[i]][j].st;
                          ans = max(ans, top + ek);
                     }
                }
                else{
                     ll vyer = yer[v[i]][u[i]];
                     for(ll j = 0; j < min(k - 2, (ll)g[v[i]].size());
j++){
                          if(g[v[i]][j].nd.st == u[i]){
                               geldi = 1;
                               continue;
                          }
                          ll bak = k - (j - geldi + 1 + 2) - 1;
                          ll ek = 0;
                          if(bak >= 0){
                               if(bak >= vyer){
                                    if(a[v[i]] < 0){
                                         ek = pre[u[i]][vyer - 1];
                                    } else{
                                         ek = pre[u[i]][min(bak + 1,
(ll)pre[u[i]].size() - 1)] - a[v[i]];
```

```
                                    }
                                }else
                                    ek      =      pre[u[i]][min(bak,
(ll)pre[u[i]].size() - 1)];
                                }
                                top += g[v[i]][j].st;
                                ans = max(ans, top + ek);
                            }
                        }
                    }
                }
                printf("%lld\n", ans);
                return 0;
}
```