**Анализ**

За улеснение ще въведем понятието, че едно дърво (*h*’, *c*’) е по-малко от друго дърво (*h*’’, *c*’’), ако *h*’ < *h*’’ или *h*’ = *h*’’ и *c*’ < *c*’’. Освен това ще казваме, че две дървета са равни, ако имат съвпадаща височина и цвят. Едно важно наблюдение е, че това сравнение е хубаво, защото е транзитивно и за всеки две неравни дървета, едното ще е по-малко. В решенията за до 70 точки може да приемаме, че дърветата нямат цвят, защото той не оказва влияние там.

Решение за 26 точки. Тук е предвидено да се направи пълно изчерпване. Фиксира се всяко дърво и се обхождат дърветата вляво от фиксирането, като се гледа минималния индекс на по-малко или равно дърво. Същото се прави и вдясно. Така сложността е .

Решение за 70 точки. Има два подхода, които могат да се направят. Единият оптимизира предната идея. Можем да се движим малко по-малко надясно и в сегментно дърво да отбелязваме индексите за съответните височини, които виждаме. Така за фиксирано дърво можем с една заявка да намерим минималния индекс на дърво вляво, което е по-ниско или със същата височина. На втори пас можем да направим същото само че в посока отдясно-наляво. Другият подход е по-важен, защото той ще ни доведе до пълната идея за решение. Може да се забележи, че наредбата на дърветата не е от особено значение, а това, което е важно, е наредбата на дърветата по-големина. Нека ги сортираме в нарастващ ред. Тогава ако обхождаме тази наредба, преди да стигнем дърво, ще сме обходили всички по-ниски дървета. Това означава, че е достатъчно да пазим минимален и максимален индекс, който срещнем. Един частен случай е ако има дървета с равна височина. Най-просто е да обхождаме последователностите в сортираната наредба от равни дървета и така да намираме минималния и максималния индекс за всяко дърво. Очакваната сложността е .

Решение за 83 точки. Малко трябва да оптимизираме последния подход. Тук ще се възползваме, че цветовете са малки. По този начин можем да направим *count sort*, като кодираме дърво (*h*, *c*) по следния начин: *h*\*11 + *c*. Лесно може да се види, че тези кодове идеално съответстват на неравенствата между две дървета. Така като превърнахме двете характеристики в едно число, което ще е по-малко от 1,2.106 ще използваме *count sort* и след това вече описания алгоритъм по сортираната наредба. За да се хванат тези точки трябва сортировката да е написана добре. Може да се забележи, че няма нужда да правим реално сортиране, а е достатъчно за всеки един код да намерим минималния и максималния индекс, където се среща такова дърво. Сложността тук ще е .

Решение за 100 точки. Когато имаме да сортираме наредени двойки, един стандартен подход е да направим *radix sort*, което е еквивалентно на два *count sort*-а. Отново трябва да се напишат добре, т.е. без използване на вектори. Друг подход би бил да кодираме двете числа в едно, но така ще работим с *long long* и поради ограниченото време такъв подход надали би успял да хване тестовете. Останалата част от алгоритъма няма промяна. Крайната сложност е .

*Автор: Илиян Йорданов*