Connect

Тагове	На пълното решение	На частичните решения
	Минимално покриващо дърво	Обхождания на графи Пълно изчерпване Динамично по подмножества

Анализ

Задачата имаше много подзадачи, които съсигурност имат интересни решения, но ще спестя обясняването на тези подзадачи, които не са свързани с пълното решение.

Решение за $N \leq 1\ 000$

За да направим такова решение, трябва малко да трансформираме задачата. Тъй като задачата е дадена в С група е силно вероятно да се сведе до някой от познатите ни алгоритми в графи. Тъй като в задачата няма нищо общо свързано с пътища, не е логично да използваме Дейкстра и подобни алгоритми. Какви алгоритми знаем за свързаност в графи? И то не каква да е свързаност, а минимална такава. Нека помислим дали може да ползваме алгоритъм за минимално покриващо дърво. Тогава е логично да конструираме граф, където теглото на всяко ребро да е цената да се използва в крайния граф. Задачата така ни се свежда до това да изберем такова множество от ребра от този граф, което има минимална сумарна цена, чиито ребра са ни достатъчни за да бъде графа свързан. Това е точно дефиницията на минималното покриващо дърво. По-формално казано, решението ни ще се състои в следните стъпки:

Нека разгледаме пълен граф, където теглото на реброто между върхове u и v е:

- 0, ако първоначално са били свързани.
- $|c_u - c_v|$ в противен случай.

Нека разгледаме минималното покриващо дърво на този граф. Лесно може да се види, че ребрата, които участват свързват графа и са с минимална цена. Ние намираме минималното покриващо дърво, примерно с алгоритъма на Крускал, като по този начин намираме отговора.

Постигната сложност: $O(N^2 \log_2 N + M)$

Имплементация: `connect_39p.cpp`

П.П. Точките за решението може да ви изглеждат малко, но силно вероятно състезател с тази идея ще стигне до $52/63$ точки, та и повече.

Решение за $N \leq 100\ 000$

Нека разгледаме така хубавия пълен граф. Чисто интуитивно може да се забележи, че има много ненужни ребра в него. Например в една свързана компонента от K върха в първоначалния граф ще има $\frac{K(K-1)}{2}$ ребра с цена 0 в нея, като са ни нужни само $K-1$. Всеки състезател може да се пробва да разглежда подобни частни случаи и да намалява нужните ребра, но не мисля, че с това той би могъл да реши задачата за 100 точки, поне ще се постарая с тестовете да не го позволя ☺.

Друго интуитивно нещо, което състезател може да се сети е да си сортира височините на върховете. Нека си преномерираме върховете спрямо височината им, като връх 1 ще е този с най-малка височина, а връх N ще е този с най-голяма височина.

За пълното решение, трябва да се забележи, че няма смисъл да се построява ребро между върхове, които не са със съседни номера. Нека допуснем, че е оптимално да се построи директно ребро между върхове u и v ($1 \leq u < v-1 \leq N$), които не са свързани. Нека вместо това построим ребра между върхове с номера $(u, u+1); (u+1, u+2); \dots; (v-1, v)$. Очевидно, като построим тези ребра, пак ще свържем компонентите на u и v. Тогава цената за тези ребра ще $(c_v - c_{v-1}) + (c_{v-1} - c_{v-2}) + \cdots + (c_{u+1} - c_u)$ и след като разкрием скобите се получава, че цената им ще е точно $c_v - c_u$, което е равно и на цената на директното ребро между u и v. Следователно достигаме до противоречие.

Нека използваме Крускал. Чрез DSU първоначално свързваме върховете чрез ребрата от входа. След това сортираме C, сортираме и ребрата, образувани от върхове със съседни номера. След това намираме и минималното покриващо дърво. Красиво, а?

Постигната сложност: $O(N \log_2 N + M)$

Имплементация: `connect_100p.cpp`

П.П. Оригинално задачата беше планирана за откритото първенство на София по информатика, където цената за построяване на ребро между върхове с номера i и j щеше да бъде $(i-j)^2$. Тогава с подобни разсъждения може да се види, че отговорът е $K-1$, където K е броят на компонентите в графа. Втората подзадача на текущата задача е абсолютно същата като $(i-j)^2$, сложих я като подсказка за пълното решение. Дано сте я **e**n**joi**-нали ☺.

Автор: Борис Михов