1029. Two City Scheduling

25 March 2022 04:27 PM

A company is planning to interview 2n people. Given the array costs where $costs[i] = [aCost_i, bCost_i]$, the cost of flying the i^{th} person to city a is $aCost_i$, and the cost of flying the i^{th} person to city b is $bCost_i$.

Return the minimum cost to fly every person to a city such that exactly n people arrive in each city.

Example 1:

Input: costs = [[10,20],[30,200],[400,50],[30,20]]
Output: 110

Explanation:

The first person goes to city A for a cost of 10. The second person goes to city A for a cost of 30. The third person goes to city B for a cost of 50. The fourth person goes to city B for a cost of 20.

The total minimum cost is 10 + 30 + 50 + 20 = 110 to have half the people interviewing in each city.

Example 2:

Input: costs = [[259,770],[448,54],[926,667],

[184,139],[840,118],[577,469]]

Output: 1859

Example 3:

Input: costs = [[515,563],[451,713],[537,709],

[343,819],[855,779],[457,60],[650,359],[631,42]]

Output: 3086

Understand the questions

* 2N folks

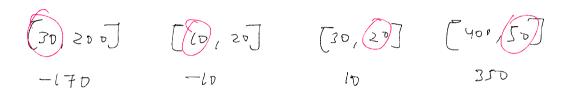
* Divide it into 2 groups Visiting city A Vs city B

& Both should have N folks

a minimum cost to fly every person that exactly in people arrive in each liby.

cat A B	loss visiting city city city	
[10, 20]	lo	-ve > it's significant that you will get
[30, 200]	L70	benefited if you vist city A value B
[00,50]	350	tre > Herr it's significant that you will
[30,20]	(0	get this much gain instead of loss if
		you visit Brather than A

we sort now, (ii) A - cly P)



Sort the array in the basic of loss of visiting by cost A - cost B

```
class Solution {
   public int twoCitySchedCost(int[][] costs) {
// sort them in basic of loss of city a and city b
//a[0], a[1] -- price for city A, and B for first candidate
//b[0], b[1] -- price for city A and B for 2nd candidate
//a[0] - a[1] = cost saving on sending first candidate to city A instead of B
//b[0] - b[1] = cost saving on sending 2nd candidate to city A instead of city B
        Arrays.sort(costs, (a,b) -> {
           return (a[0] - a[1]) - (b[0] - b[1]);
        });
        int total = 0;
        for(int i = 0; i < costs.length; i++){
            if(i < costs.length / 2){
                                               first had after sorting roll be city A
                // select city A
                total += costs[i][0];
            } else {
} else {
                // select city B
                                            Frest holf will be ally B
                total += costs[i][1];
        return total;
}
```

$$T(=) o(n \log n)$$

$$S(=) o(1)$$