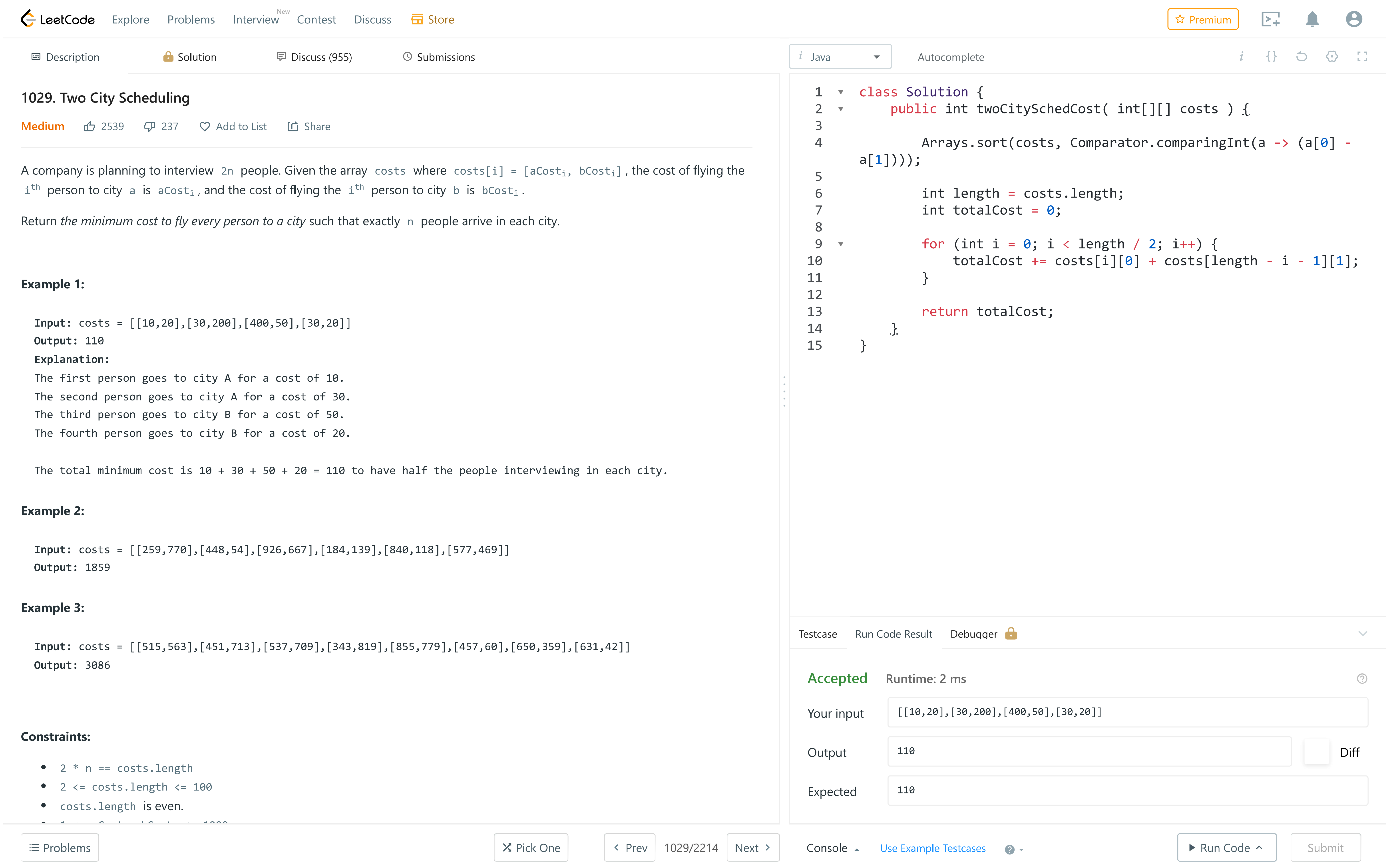

1029. Two City Scheduling

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
```

Constraints:

- `2 * n == costs.length`
- `2 <= costs.length <= 100`
- `costs.length` is even.

```java
class Solution {
    public int twoCitySchedCost( int[][] costs ) {

        Arrays.sort(costs, Comparator.comparingInt(a -> (a[0] -
a[1])));

        int length = costs.length;
        int totalCost = 0;

        for (int i = 0; i < length / 2; i++) {
            totalCost += costs[i][0] + costs[length - i - 1][1];
        }

        return totalCost;
    }
}
```

Accepted Runtime: 2 ms

Your input `[[10,20],[30,200],[400,50],[30,20]]`

Output `110` Diff

Expected `110`