5650. Minimize Hamming Distance After Swap Operations

My Submissions (/contest/weekly-contest-223/problems/minimize-hamming-distance-after-swap-operations/submissions/)

Back to Contest (/contest/weekly-contest-223/)

You are given two integer arrays, `source` and `target`, both of length `n`. You are also given an array `allowedSwaps` where each `allowedSwaps[i] = [a_i, b_i]` indicates that you are allowed to swap the elements at index `a_i` and index `b_i` **(0-indexed)** of array `source`. Note that you can swap elements at a specific pair of indices **multiple** times and in **any** order.

The **Hamming distance** of two arrays of the same length, `source` and `target`, is the number of positions where the elements are different. Formally, it is the number of indices `i` for `0 <= i <= n-1` where `source[i] != target[i]` **(0-indexed)**.

Return *the **minimum Hamming distance** of* `source` *and* `target` *after performing **any** amount of swap operations on array* `source`.

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

Example 1:

```
Input: source = [1,2,3,4], target = [2,1,4,5], allowedSwaps = [[0,1],[2,3]]
Output: 1
Explanation: source can be transformed the following way:
- Swap indices 0 and 1: source = [2,1,3,4]
- Swap indices 2 and 3: source = [2,1,4,3]
The Hamming distance of source and target is 1 as they differ in 1 position: index 3.
```

Example 2:

```
Input: source = [1,2,3,4], target = [1,3,2,4], allowedSwaps = []
Output: 2
Explanation: There are no allowed swaps.
The Hamming distance of source and target is 2 as they differ in 2 positions: index 1 and index 2.
```

Example 3:

```
Input: source = [5,1,2,4,3], target = [1,5,4,2,3], allowedSwaps = [[0,4],[4,2],[1,3],[1,4]]
Output: 0
```

Constraints:

- `n == source.length == target.length`
- `1 <= n <= 10^5`
- `1 <= source[i], target[i] <= 10^5`
- `0 <= allowedSwaps.length <= 10^5`
- `allowedSwaps[i].length == 2`
- `0 <= a_i, b_i <= n - 1`
- `a_i != b_i`

JavaScript

```
/**
 * @param {number[]} source
 * @param {number[]} target
 * @param {number[][]} allowedSwaps
 * @return {number}
 */
var minimumHammingDistance = function(source, target, allowedSwaps) {

};
```