Problem 3587: Minimum Adjacent Swaps to Alternate Parity

Problem Information

Difficulty: Medium
Acceptance Rate: 42.00%
Paid Only: No
Tags: Array, Greedy

Problem Description

You are given an array `nums` of **distinct** integers.

In one operation, you can swap any two **adjacent** elements in the array.

An arrangement of the array is considered **valid** if the parity of adjacent elements **alternates** , meaning every pair of neighboring elements consists of one even and one odd number.

Return the **minimum** number of adjacent swaps required to transform `nums` into any valid arrangement.

If it is impossible to rearrange `nums` such that no two adjacent elements have the same parity, return `-1`.

Example 1:

Input: nums = [2,4,6,5,7]

Output: 3

Explanation:

Swapping 5 and 6, the array becomes `[2,4,5,6,7]`

Swapping 5 and 4, the array becomes `[2,5,4,6,7]`

Swapping 6 and 7, the array becomes `[2,5,4,7,6]`. The array is now a valid arrangement. Thus, the answer is 3.

Example 2:

Input: nums = [2,4,5,7]

Output: 1

Explanation:

By swapping 4 and 5, the array becomes `[2,5,4,7]`, which is a valid arrangement. Thus, the answer is 1.

Example 3:

Input: nums = [1,2,3]

Output: 0

Explanation:

The array is already a valid arrangement. Thus, no operations are needed.

Example 4:

Input: nums = [4,5,6,8]

Output: -1

Explanation:

No valid arrangement is possible. Thus, the answer is -1.

Constraints:

* `1 <= nums.length <= 105` * `1 <= nums[i] <= 109` * All elements in `nums` are **distinct**.

Code Snippets

C++:

```
class Solution {
public:
int minSwaps(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int minSwaps(int[] nums) {

}
}
```

Python3:

```
class Solution:
def minSwaps(self, nums: List[int]) -> int:
```