Problem 2035: Partition Array Into Two Arrays to Minimize Sum Difference

Problem Information

Difficulty: Hard
Acceptance Rate: 22.59%
Paid Only: No
Tags: Array, Two Pointers, Binary Search, Dynamic Programming, Bit Manipulation, Ordered Set, Bitmask

Problem Description

You are given an integer array `nums` of `2 * n` integers. You need to partition `nums` into **two** arrays of length `n` to **minimize the absolute difference** of the **sums** of the arrays. To partition `nums`, put each element of `nums` into **one** of the two arrays.

Return _the**minimum** possible absolute difference_.

Example 1:

![example-1](https://assets.leetcode.com/uploads/2021/10/02/ex1.png)

Input: nums = [3,9,7,3] **Output:** 2 **Explanation:** One optimal partition is: [3,9] and [7,3]. The absolute difference between the sums of the arrays is abs((3 + 9) - (7 + 3)) = 2.

Example 2:

Input: nums = [-36,36] **Output:** 72 **Explanation:** One optimal partition is: [-36] and [36]. The absolute difference between the sums of the arrays is abs((-36) - (36)) = 72.

Example 3:

![example-3](https://assets.leetcode.com/uploads/2021/10/02/ex3.png)

Input: nums = [2,-1,0,4,-2,-9] **Output:** 0 **Explanation:** One optimal partition is: [2,4,-9] and [-1,0,-2]. The absolute difference between the sums of the arrays is abs((2 + 4 + -9) - (-1 + 0 + -2)) = 0.

Constraints:

* `1 <= n <= 15` * `nums.length == 2 * n` * `-107 <= nums[i] <= 107`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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