Problem 1013: Partition Array Into Three Parts With Equal Sum

Problem Information

Difficulty: Easy
Acceptance Rate: 42.35%
Paid Only: No
Tags: Array, Greedy

Problem Description

Given an array of integers `arr`, return `true` if we can partition the array into three **non-empty** parts with equal sums.

Formally, we can partition the array if we can find indexes `i + 1 < j` with `(arr[0] + arr[1] + ... + arr[i] == arr[i + 1] + arr[i + 2] + ... + arr[j - 1] == arr[j] + arr[j + 1] + ... + arr[arr.length - 1])`

Example 1:

Input: arr = [0,2,1,-6,6,-7,9,1,2,0,1] **Output:** true **Explanation:** 0 + 2 + 1 = -6 + 6 - 7 + 9 + 1 = 2 + 0 + 1

Example 2:

Input: arr = [0,2,1,-6,6,7,9,-1,2,0,1] **Output:** false

Example 3:

Input: arr = [3,3,6,5,-2,2,5,1,-9,4] **Output:** true **Explanation:** 3 + 3 = 6 = 5 - 2 + 2 + 5 + 1 - 9 + 4

Constraints:

* `3 <= arr.length <= 5 * 104` * `-104 <= arr[i] <= 104`

Code Snippets

C++:

```
class Solution {
public:
bool canThreePartsEqualSum(vector<int>& arr) {

}
};
```

Java:

```
class Solution {
public boolean canThreePartsEqualSum(int[] arr) {

}
}
```

Python3:

```
class Solution:
def canThreePartsEqualSum(self, arr: List[int]) -> bool:
```