454. 4Sum II

Given four integer arrays `nums1`, `nums2`, `nums3`, and `nums4` all of length `n`, return the number of tuples `(i, j, k, l)` such that:

- `0 <= i, j, k, l < n`
- `nums1[i] + nums2[j] + nums3[k] + nums4[l] == 0`

Example 1:

Input: nums1 = [1,2], nums2 = [-2,-1], nums3 = [-1,2], nums4 = [0,2]
Output: 2
Explanation:
The two tuples are:
1. (0, 0, 0, 1) -> nums1[0] + nums2[0] + nums3[0] + nums4[1] = 1 + (-2) + (-1) + 2 = 0
2. (1, 1, 0, 0) -> nums1[1] + nums2[1] + nums3[0] + nums4[0] = 2 + (-1) + (-1) + 0 = 0

Example 2:

Input: nums1 = [0], nums2 = [0], nums3 = [0], nums4 = [0]
Output: 1

```
def fourSumCount(self, A: List[int], B: List[int], C: List[int], D:
List[int]) -> int:
        # hashTable = {}
        # for a in nums1:
        #     for b in nums2:
        #         hashTable[a+b] = hashTable.get(a+b,0)+1
        # count = 0
        # for c in nums3:
        #     for d in nums4:
        #         if -c-d in hashTable:
        #             count += hashTable[-c-d]
        # return count
        sums = collections.Counter(c+d for c in C for d in D)
        return sum(sums.get(-(a+b), 0) for a in A for b in B)
```