题目：

Given four lists A, B, C, D of integer values, compute how many tuples `(i, j, k, l)` there are such that `A[i] + B[j] + C[k] + D[l]` is zero.

To make problem a bit easier, all A, B, C, D have same length of N where 0 ≤ N ≤ 500. All integers are in the range of -2^{28} to $2^{28} - 1$ and the result is guaranteed to be at most $2^{31} - 1$.

Example:

```
Input:
A = [ 1, 2]
B = [-2,-1]
C = [-1, 2]
D = [ 0, 2]

Output:
2

Explanation:
The two tuples are:
1. (0, 0, 0, 1) -> A[0] + B[0] + C[0] + D[1] = 1 + (-2) + (-1) + 2 = 0
2. (1, 1, 0, 0) -> A[1] + B[1] + C[0] + D[0] = 2 + (-1) + (-1) + 0 = 0
```

Subscribe to see which companies asked this question.

1.时间：O（N^2）;空间：O（N^2）

```
class Solution {
public:
    int fourSumCount(vector<int>& A, vector<int>& B, vector<int>& C,
```

```
vector<int>& D) {
        std::sort(A.begin(), A.end());
        std::sort(B.begin(), B.end());
        std::sort(C.begin(), C.end());
        std::sort(D.begin(), D.end());/* 排序后可以加快速度 */

        std::unordered_map<int, int> hashTable;
        for (int i = 0; i < A.size(); ++i){
            for (int k = 0; k < B.size(); ++k){
                int sum = A[i] + B[k];
                hashTable[sum]++;
            }
        }
        int result = 0;
        for (int i = 0; i < C.size(); ++i){
            for (int k = 0; k < D.size(); ++k){
                int sum = -1 * (C[i] + D[k]);
                if (hashTable.find(sum) != hashTable.end()){
                    result += hashTable[sum];
                }
            }
        }
```

```
        return result;

    }

};
```