Exercise 1.4.15: Faster 3-sum. As a warmup, develop an implementation TwoSumFaster that uses a *linear* algorithm to count the pairs that sum to zero after the array is sorted (*instead of* the binary-search-based *linearithmic* algorithm). Then apply a similar idea to develop a *quadratic* algorithm for the 3-sum problem.

Solution:

```
//PseudoCode: Faster 3-sum
//Assume arrays already sorted.
//Find linearithmic solution for pairs that sum to 0

function Check_two_faster_sum(arr []):
    int a = 0;
    int b = arr.length-1;

    int count = 0;
    while(b>a): //first find the pairs that result in 0 initially, then use again after
        if(arr[a] + arr[b] > 0):
            b--

        else if(arr[a] + arr[b] < 0):
            a++

        else:
            count++

            //compare all following elements w/array[end]
            tempIndex = start + 1

            while(tempIndex < end && (arr[tempIndex] + arr[end] == 0)):
                count++;
                tempIndex++;

            //compare all prev elements w/array[start]
            tempIndex = end-1
            while(tempIndex>start && (arr[start] + array[tempIndex] == 0)){
                count++
                tempIndex--
            }

        start++;
        end--;
    return count;
```

```
//PseudoCode: Faster 3-sum
//Assume arrays already sorted.
//Quadtratic soln assuming integers are distinct.
//The code below sorts tge array, then does n(n-1)/2 binary searches
//This takes time proportional to log n, for a total runtime of (n^2)logn
//If duplicates occur or if we do bruteforce, then time proportional to O(n^3)

/*
A pair a[i] & a[j] is part of a triple( 3 # set) that sums to
0 if and only if value @ -(a[i] + a[j]) is in the array. This value would
represent the third value. We can do this using Binary Search
*/

//count truokes that sum to 0
int count_3_sums(a[]){


    N = a.length
    cnt = 0;

    for i in range(N):
        for(int j = i+1; j<N; j++):
            if(BinSearch.indexOf(a,-(a[i] + a[j]))> j):
                cnt++;
    return cnt;
}
```