Count Triplets in C++

```
#include <iostream>
#include <algorithm>
using namespace std;

class CountTheTriplets {
public:
    static int countTriplets(int arr[], int n)
{
        // Sort the array
        sort(arr, arr + n);
        int count = 0;

        // Traverse the array from the end to
find triplets
        for (int i = n - 1; i >= 2; i--) {
            int left = 0, right = i - 1;

            // Two pointers technique to find
triplets
            while (left < right) {
                if (arr[left] + arr[right] ==
arr[i]) {
                    // If valid triplet is found
                    count++;
                    left++;
                    right--;
                } else if (arr[left] + arr[right] <
arr[i]) {
                    // Move left pointer to
increase the sum
                    left++;
                } else {
                    // Move right pointer to
decrease the sum
                    right--;
                }
            }
        }

        return count;
    }
};

int main() {
    // Hardcoded input
    int n = 6;
    int arr[] = {1, 3, 5, 2, 7, 4};

    // Call the countTriplets method to
count triplets
    int result =
CountTheTriplets::countTriplets(arr, n);

    // Print the result
    cout << "Number of triplets: " << result
<< endl;

    return 0;
}
```

Count the number of **triplets (i, j, k)** in the array such that:

arr[i] + arr[j] == arr[k]

Where i, j, and k are **distinct indices**.

✔ **Input Array:**

arr[] = {1, 3, 5, 2, 7, 4}

n = 6

🔄 **After Sorting:**

arr[] = {1, 2, 3, 4, 5, 7}
↑ ↑ ↑ ↑ ↑ ↑
0 1 2 3 4 5 (indexes)

🧪 **Dry Run Table:**

i (arr[i])	left	right	arr[left] + arr[right]	Comparison	Action	Count
5 (7)	0	4	1 + 5 = 6	< 7	left++ → left=1	0
	1	4	2 + 5 = 7	== 7 → **Triplet found!**	count++, left++, right--	1
	2	3	3 + 4 = 7	== 7 → **Triplet found!**	count++, left++, right--	2
4 (5)	0	3	1 + 4 = 5	== 5 → **Triplet found!**	count++, left++, right--	3
	1	2	2 + 3 = 5	== 5 → **Triplet found!**	count++, left++, right--	4
3 (4)	0	2	1 + 3 = 4	== 4 → **Triplet found!**	count++, left++, right--	5
	1	1	loop ends			
2 (3)	0	1	1 + 2 = 3	== 3 → **Triplet found!**	count++, left++, right--	6

🎯 **Final Output:**

Number of triplets: 6

✔ **Triplets Found:**

- (2, 5) → 2 + 5 = 7

	• (3, 4) → 3 + 4 = 7 • (1, 4) → 1 + 4 = 5 • (2, 3) → 2 + 3 = 5 • (1, 3) → 1 + 3 = 4 • (1, 2) → 1 + 2 = 3
Number of triplets: 6	