Split Array

Problem Description: You are given with an input array, then you have to check whether this array can be split into two subarrays having equal sum such that:

- All elements in the input, which are divisible by 5 should be in same group.
- All elements in the input, which are divisible by 3 (but not divisible by 5) should be in other group.
- Elements which are neither divisible by 5 nor by 3, can be put in any group.

Return true, if array can be split according to the above rules, else return false.

How to approach:

We can use a recursive approach by keeping left sum and right sum to maintain two different subarrays. Left sum is for multiples of 5 and right sum is for multiples of 3 (which are not multiples of 5) and the elements which are neither divisible by 5 nor by 3 can lie in any group satisfying the equal sum rule (include them in left sum and right sum one by one). And then check whether the left sum is equal to the right sum in the base case. If equal the you can print true otherwise print false.

CODE:

```
bool check(int *input, int size, int startIndex, int lSum, int rSum) {
    // If we reach the end
    if(startIndex == size) {
        return lSum == rSum;
    }
    // If number is divisible by 5 add it to the left sum
    if(input[startIndex] % 5 == 0) {
        lSum += input[startIndex];
    }
    // If number is divisible by 3 not by 5 add it to the right sum
    else if(input[startIndex] % 3 == 0) {
        rSum += input[startIndex];
    }
    else {
    //Try adding them in both arrays(one by one) and check whether it
//is satisfying the sum condition or not.
        bool leftAns = check(input, size, startIndex+1,
                                lSum+input[startIndex], rSum);
        bool rightAns = check(input, size, startIndex+1,
                                 lSum, rSum + input[startIndex]);
        return leftAns || rightAns;
    }
    // Cases when either divisible by 5 or 3.
    return check(input, size, startIndex + 1, lSum, rSum);
}

bool splitArray(int *input, int size) {
    return check(input, size, 0, 0, 0);

}
```