Calculate the Sum of Array using recursion

The provided C++ program calculates the sum of elements in an array using a recursive approach.

Recursive function to calculate the sum of elements in an array

1. Inside the **sumOfArray** function, there are three cases:
   - Base Case 1: If the **size** of the array is 0, it means the array is empty, and the sum is 0. In this case, the function returns 0.
   - Base Case 2: If the **size** of the array is 1, it means the array has only one element, and the sum is that element. In this case, the function returns the value of **arr[0]**.
   - Recursive Case: If the array has more than one element, the function calculates the sum of the array by adding the first element **arr[0]** with the result of the recursive call to **sumOfArray** with the rest of the elements in the array (i.e., **arr + 1**) and the reduced size of the array (**size - 1**).

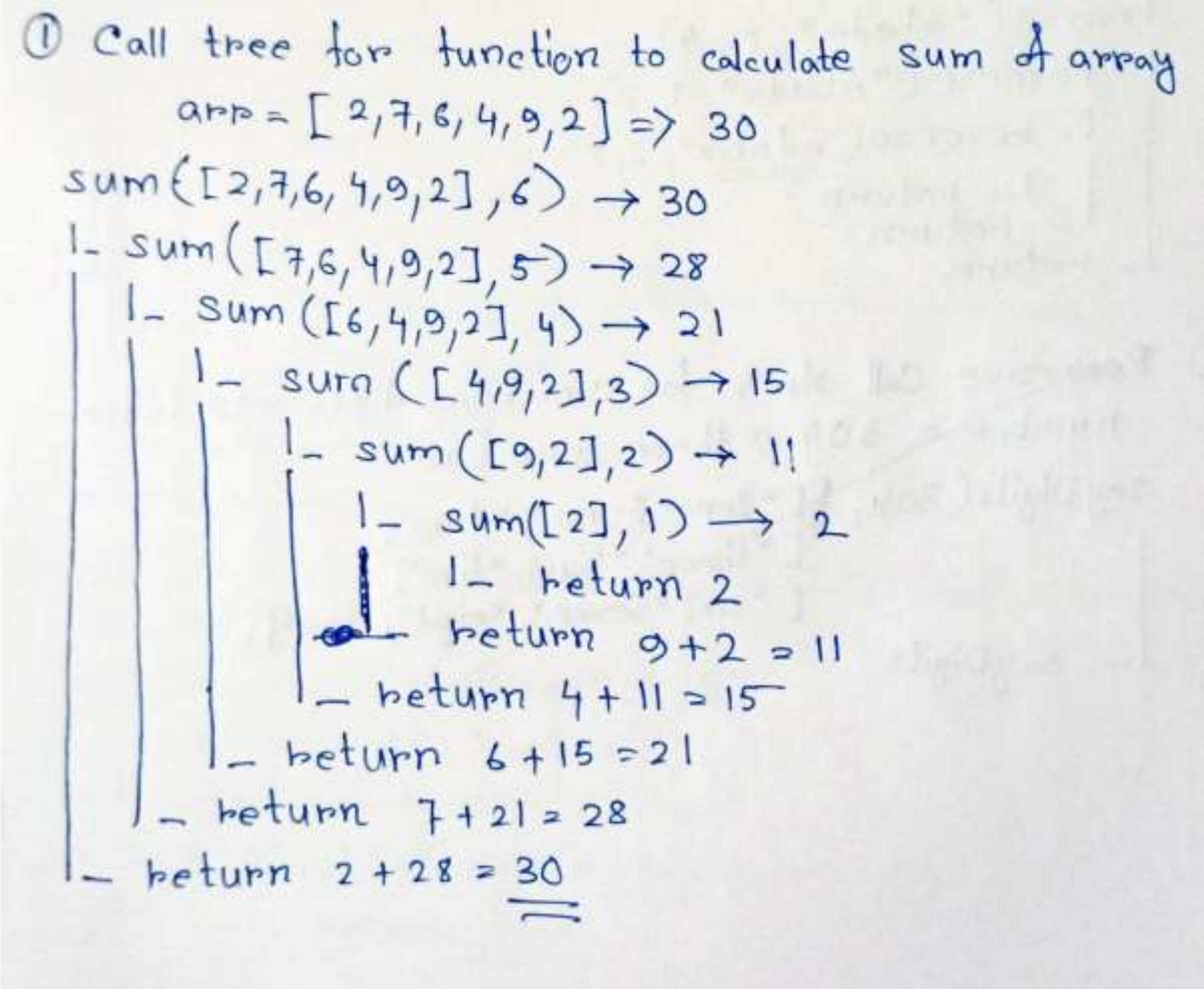
Time Complexity:

The time complexity of the sumOfArray function is O(n), where n is the size of the array. This is because in each recursive call, the function processes one element of the array, and it makes n recursive calls in the worst case, one for each element.

Space Complexity:

The space complexity of the program is O(n), where n is the size of the array. This is because the recursive calls in the sumOfArray function create new frames on the call stack, and in the worst case, there can be n recursive calls, leading to O(n) space consumption on the call stack. Additionally, the **arr** array has a size of n, contributing to the space complexity as well.

Recursive call stack of the approach:



① Call tree for function to calculate sum of array

```
        arr = [2,7,6,4,9,2] => 30
  sum([2,7,6,4,9,2],6) → 30
  |- sum([7,6,4,9,2],5) → 28
  |   |- sum([6,4,9,2],4) → 21
  |   |   |- sum([4,9,2],3) → 15
  |   |   |   |- sum([9,2],2) → 11
  |   |   |   |   |- sum([2],1) → 2
  |   |   |   |   |   |- return 2
  |   |   |   |   |- return 9+2 = 11
  |   |   |   |- return 4 + 11 = 15
  |   |   |- return 6 + 15 = 21
  |   |- return 7 + 21 = 28
  |- return 2 + 28 = 30
```