Split array problem

First of all, the problem says that “the average value of B is equal to the average value of C”. Following some simple steps, it can be proven that the average of B and C will also be equal to the average of A:

Seeing that “B and C are both non-empty”, it means that if we denote the length of B as ‘k’ and the length of A as ‘n’, we have that 0 < k < n.

We start from average(B) = average(C) and write it as sum(B)/length(B) = sum(C)/length(C) (Assume A is ordered in such a way that the first k elements are the elements of B).

$$\frac{\sum_{i=1}^{k} A_i}{k} = \frac{\sum_{i=k+1}^{n} A_i}{n-k}$$

$$(n-k) * \sum_{i=1}^{k} A_i = k * \sum_{i=k+1}^{n} A_i$$

$$n * \sum_{i=1}^{k} A_i - k * \sum_{i=1}^{k} A_i = k * \sum_{i=k+1}^{n} A_i$$

$$n * \sum_{i=1}^{k} A_i = k * \left(\sum_{i=1}^{k} A_i + \sum_{i=k+1}^{n} A_i \right)$$

$$n * \sum_{i=1}^{k} A_i = k * \sum_{i=1}^{n} A_i$$

$$\frac{\sum_{i=1}^{k} A_i}{k} = \frac{\sum_{i=1}^{n} A_i}{n}$$

So we have that sum(B)/length(B) = sum(A)/length(A), which means that average(B) = average(A).

Explaining the algorithm:

The problem at hand is to construct B in such a way that its average is equal to the average of A and length of B is between 0 and the length of A (non-inclusive). We will generate all possibilities of choosing elements from array A for list B.

To find the average of B, we will need the sum and the number of elements of B, so we create a boolean matrix for that, named “existsB”, which will have only “false” values on declaration. Element existsB[s][i] will be set to true if there exists a way of choosing i elements from A with the sum s.

The element existsB[0][0] will be set to “true” before generating the rest of the matrix, so that the algorithm can add the first element of B.

In order to generate the matrix we will need to go through all elements of A. For each element x of A, we will go through the “existsB” matrix and update the positions in which we add x to true, in the following way:

For an element x of A, we go through the rows of matrix “existsB” with an index s (the rows are for the sum) in decreasing order so that we avoid adding x multiple times. On each row we go through the columns with an index j (columns are for number of elements that add up to that sum) in decreasing order, for the same reason as above, and check if the position is marked as true. If it is, we mark existsB[s + x][j + 1] with true (we add x to the list found until then).

Every time we mark a new position of “existsB” we check if we have a solution, which is the following case: length(B) is smaller than length(A) (and it will always be greater than 0 because we just added an element) and sum(A)*length(B) = sum(B)*length(A) (their averages are the same). If we don’t find a solution and the matrix has been completely generated, then there is no solution for the given array A.