Algorithm Design Practice Problems

I have already posted an assignment on algorithm design problems. In addition to those assignment questions, you can practice the following problems as well for brainstorming.

Q#1: Given two sorted sets of size “N” each represented in the form of two arrays “A[1…N] and B[1…N]”. The task is to get the intersection and union of these sets. You can store the intersection/union in a resultant array R[]. The union is not necessarily in sorted order. Provide the pseudo code of the most efficient approach to solve this problem using the divide and conquer approach. (Is there any other approach to get a more efficient solution than divide and conquer).

Input: A[] = {3, 5, 7, 13, 19, 21}; B[] = {1,2, 5, 8, 13, 15}; **Output**(intersection of A and B) = { 5,13} Union of A and B = {3,5,7,13,19,21,1,2,8,15}

Q#2: Given an array of integers having both +ve and -ve values. The task is to find two elements of the array having sum closest to zero.

A[] = {1, 60, -10, -80, 70, 85} => Answer: (-80, 85)

Q#3: Given an array of integers of size “N” and a target value.

- Determine whether there exists any subarray having sum equals to the target value and return either true or false.
- Now assume that we are sure that there exists a sub array having sum equals to the target value. Your task is to return the sub-array, or you can simply return the starting or ending indices of that sub array.

Q#4: Given an array of integers having both +ve and -ve values of size “N”. Your task is to determine the longest subarray (or starting and ending indices of longest subarray) having sum equals to zero.

- You can also think of this problem as finding the longest subarray having sum equals to the target value.

Q#5: Given an array of integers “A[]” of size “N”. The task is to return a resultant array “R[]” whose each index “i” should hold the count of smaller values on the right side of A[i].

A[] = {12, 3, 5, 2, 4, 15, 10} There are total 6 elements on the right side of 12 i.e., {3, 5, 2, 4, 15, 10} out of these 6 elements, 5 elements are smaller than 12 so R[1] = 5 similarly the count of smaller elements on the right side of A[i] will be calculated and the final result of R[] will be:

Output: R[] = {5, 1, 2, 0, 0, 1, 0}