Problem 1. Recall that, in merge sort, the merging step combines two sorted lists \(A_1, A_2\) into one sorted list \(A\). Suppose that \(A_1\) and \(A_2\) are \((1, 8, 17, 23, 35, 83)\) and \((3, 5, 15, 28, 56)\), respectively. Describe the contents of \(A\) right before the moment when 23 enters \(A\).

Problem 2. Let \(S\) be a set of \(n\) integers stored in an array (not necessarily sorted). Design an algorithm to find the 10 largest integers in \(S\) (by creating a separate array of length 10 storing those integers). Your algorithm must finish in \(O(n)\) time.

Problem 3. Let \(S\) be a set of \(n\) integers stored in an array (not necessarily sorted). Let \(k \geq 1\) be an integer. Design an algorithm to find the \(k\) largest integers in \(S\) (by creating a separate array of length \(k\) storing those integers). Your algorithm must finish in \(O(n \log n)\) time.

Problem 4. Let \(S_1\) and \(S_2\) be two disjoint sets of integers, i.e., \(S_1 \cap S_2 = \emptyset\). We know that \(|S_1| = |S_2| = n\) (i.e., each set has \(n\) integers). Each set is stored in an array of length \(n\), where its integers are sorted in ascending order. Let \(k \geq 1\) be an integer. Design an algorithm to find the \(k\) smallest integers in \(S_1 \cup S_2\) in \(O(k)\) time.

Problem 5. Let \(S_1\) and \(S_2\) be two sets of integers (they are not necessarily disjoint). We know that \(|S_1| = |S_2| = n\) (i.e., each set has \(n\) integers). Each set is stored in an array of length \(n\), where its integers are sorted in ascending order. Let \(k \geq 1\) be an integer. Design an algorithm to find the \(k\) smallest integers in \(S_1 \cap S_2\) in \(O(n)\) time.

Problem 6 (Two Sum, Again). Let \(S\) be a set of \(n\) integers stored in an array of length \(n\). You are also given a value of \(v\). Design an algorithm to determine whether \(S\) has two integers that add up to \(v\). Your algorithm should terminate in \(O(n \log n)\) time.