1. Is an array that is in sorted order a min-heap?

2. What is the effect of calling MAX-HEAPIFY(A, i) for \( i > \text{A.heap-size}/2 \)?

3. Give an \( O(n \lg k) \)-time algorithm to merge \( k \) sorted lists into one sorted list, where \( n \) is the total number of elements in all the input lists. (Hint: Use a min-heap for \( k \)-way merging.)

4. Show that the running time of QUICKSORT is \( \Theta(n^2) \) when the array \( A \) contains distinct elements and is sorted in decreasing order.

5. When RANDOMIZED-QUICKSORT runs, how many calls are made to the random number generator RANDOM in the worst case? How about in the best case? Give your answer in terms of \( \Theta \)-notation.

6. We can improve the running time of quicksort in practice by taking advantage of the fast running time of insertion sort when its input is nearly sorted. Upon calling quicksort on a subarray with fewer than \( k \) elements, let it simply return without sorting the subarray. After the top-level call to quicksort returns, run insertion sort on the entire array to finish the sorting process. Argue that this sorting algorithm runs in \( O(nk + n \lg(n/k)) \) expected time. How should we pick \( k \), both in theory and in practice?