1. Show that a set of $n$ numbers ($n$ a multiple of $p$, $p$ a power of 2) can be partitioned in $O(n \log p)$ time into $p$ sets $S_1, S_2, ..., S_p$ of size $\frac{n}{p}$ such that every element $S_i \leq$ every element of $S_{i+1}$. Note: make sure your algorithm doesn’t take $\Theta(np)$ time, which is what you get if you search sequentially for element $\frac{n}{p}$, $2\frac{n}{p}$, $3\frac{n}{p}$, etc from the original set.

2. Let $X[1..n]$ and $Y[1..n]$ be two arrays, each containing $n$ numbers already in sorted order. Give an $O(\log n)$-time algorithm to find the median of all $2n$ elements in arrays $X$ and $Y$.

3. Consider the following problem: Describe an $O(n)$ algorithm that, given a set $S$ of $n$ distinct numbers and a positive integer $k \leq n$, determines the $k$ numbers in $S$ that are closest in value to the median of $S$. Assume that $n$ is odd (but it could be extended to work in either case).

4. You are given an array of $n$ elements, and you notice that some of the elements are duplicates. That is, they appear more than once in the array. Show how to remove all duplicates from the array in time $O(n \log n)$.

5. The nuts and bolts problem is defined as follows: You are given a collection of $n$ bolts of different widths and $n$ corresponding nuts. You are allowed to try a nut and bolt together, from which you can determine whether the nut is too large, too small, or an exact match for the bolt, but there is no way to compare two nuts together, or two bolts together. You are to match each bolt to its nut.

   (a) Devise an algorithm for the nuts and bolts problem that runs in time $O(n \log n)$ on average.

   (b) If you are allowed to compare bolts to bolts (but not nuts to nuts), describe an algorithm that runs in time $O(n \log n)$ in the worst case that doesn’t simply resort to sorting the bolts then sequentially checking nuts to match the sorted bolts.