1. The input is a sequence $x_1, x_2, \ldots, x_n$ of integers in an arbitrary order, and another sequence $a_1, a_2, \ldots, a_n$ that is a permutation of the integers from 1 to $n$. Both sequences are given as arrays. Design an $O(n \log n)$-time in-place algorithm to order the first sequence according to the order imposed by the permutation. In other words, for each $i$, $x_i$ should appear in the output in the position given in $a_i$. (20 points)

2. In class we showed using an adversary argument that any algorithm to compute the MAX and the MIN of a set of $n$ distinct numbers simultaneously using pairwise comparisons must, in the worst-case, use at least $\lceil \frac{3n}{2} \rceil - 2$ comparisons.

   (a) Use the decision tree argument (finding a lower bound on the number of possible responses, which is a lower bound on the number of leaves, and using this as a bound on the height of the tree) to develop a worst-case lower bound on the number of pairwise comparisons.

   (b) If this bound is different than $\lceil \frac{3n}{2} \rceil - 2$, explain how seemingly contradictory bounds can both be correct.

(15 points)

3. Given an array of integers $A[i], 1 \leq i \leq n$, such that for all $i$ we have $|A[i] - A[i + 1]| \leq 1$. Let $A[1] = x$ and $A[n] = y$, such that $x < y$. Design an optimal search algorithm to find $j$ such that $A[j] = z$ for a given value $z$, $x \leq z \leq y$. Use the decision tree argument to show that your algorithm is optimal. (15 points)

4. Show that the second largest element can be found with $n + \lceil \log n \rceil - 2$ comparisons in the worst case. (15 points)
5. In our linear-time selection algorithm, the inputs are divided into groups of 5. What if you used groups of 3 instead? What if used groups of 7 or larger (odd integers)? (15 points)

6. Show that $2n - 1$ comparisons are necessary in the worst case to merge two sorted lists containing $n$ elements each. (20 points)