# COSC 240, Spring 2020: Problem Set $\#2$

**Assigned:** Tuesday, 2/4/20.
**Due:** Thur, 2/13, at the beginning of class (hand in hard copy).
**Lectures Covered:** 7 to 9.
**Academic Integrity:** You can work with other people in the class but you must write up your own answers in your own words. You can also use the textbook and talk to the professor. You may not use any other resources (e.g., material found online) or talk to people outside the class about these problems. See the syllabus for details on the academic integrity policy for problem sets.

## Problems

1. Describe with pseudocode a divide-and-conquer algorithm that finds the largest value in an array $A$ of numbers. In more detail, your algorithm should take as arguments an array $A$, and two integers $p$, $r$, such that $1 \leq p \leq r \leq length(A) = n$, and return the largest value in $A[p...r]$.

2. Describe the step complexity of your solution to problem 1 with a recurrence. Solve the recurrence using the master method (show your work).

3. Consider the *pseudo-sorting* problem where you are provided an array $A$ containing $n > 1$ unique values (i.e., no two value are equal), and you must return an array $B$ that contains a permutation of these values reordered such that the $k = \lfloor n/2 \rfloor$ smallest values in $A$ are in the first $k$ positions of $B$. The order of the values in the first $k$ positions does not matter. All that matters is that the $k$ smallest values occupy these first $k$ positions.

   Describe in pseudocode a solution to this problem that runs in time $\Theta(n)$. Your answer should include a time complexity analysis. (It i not just enough to claim your algorithm runs in $\Theta(n)$ steps, you must show this to be true.) Three notes:

   - Your solution is allowed to make use of a **median-search**$(A)$ subroutine that returns the median value from $A$ and requires only $O(length(A))$ steps.
   - Your solution does *not* have to use a divide-and-conquer strategy. (In fact, it is much easier if you do not use divide-and-conquer.)
   - Your solution does *not* have to operate only within the original array $A$. It is fine to create an output array and copy values from $A$ into it.

4. Argue clearly and concisely why the comparison-based sorting algorithm lower bound does not apply to this pseudo-sorting problem.

5. Provide a clear and concise argument for why every (non-randomized) solution to pseudo-sorting requires $\Omega(n)$ steps. This does not have to be a formal mathematical argument as we used for the comparison-based sorting lower bound. But it should be convincing.