ECS122A Homework Problems #1 **Due: 4:00pm, Friday, January 15, 2016**

Be aware that your homework should be your own work. It is a matter of intellectual honesty to write your homework strictly by yourself. Using solutions from any other source is not allowed.

1. Use mathematical induction to show that when n is an exact power of 2, $T(n) = n \lg n$ is the solution of the recurrence relation

$$T(n) = \begin{cases} 2 & \text{if } n = 2 \\ 2T(\frac{n}{2}) + n & \text{if } n = 2^k \text{ for } k > 1. \end{cases}$$

2. Suppose we are comparing implementations of Insert-sort and Merge-sort on the same machine. For input of size $n = 2^k$ for $k \geq 1$, Insert-sort runs in $8n^2$ steps, while Merge-sort runs in $64n \lg n$ steps. For which value of n does Insert-sort beat Merge-sort?

3. We can express Insert-sort as a recursive procedure as follows. In order to sort $A[1...n]$, we recursively sort $A[1...n-1]$ and then insert $A[n]$ into sorted array $A[1...n-1]$.

   (a) Write the pseudocode for this recursive version of Insert-sort, name it Insert-sort-recur.

   (b) Write a recurrence for the running time of of Insert-sort-recur.

   (c) Find the solution of this recurrence relation.

   (d) Is Insert-sort-recur more expensive than Insert-sort?

4. In this exercise, we consider a Selection-sort algorithm. To sort n numbers stored in array a, we first find the smallest element of S and exchanging it with the element in $a[1]$. Then find the second smallest element of a, and exchange it with $a[2]$. Continue in this manner for the first $n - 1$ element of a.

   (a) Write a pseudocode for the Selection-sort algorithm.

   (b) Analyze the running times.

5. Given an array $s = \langle s[1], s[2], \ldots, s[n] \rangle$, and $n = 2^d$ for some $d \geq 1$. We want to find the minimum and maximum values in s. We do this by comparing elements of s.

   (a) The "obvious" algorithm makes $2n - 2$ comparisons. Explain.

   (b) Can we do it better? Carefully specify a more efficient divide-and-conquer algorithm.

   (c) Let $T(n) =$ the number of comparisons your algorithm makes. Write a recurrence relation for $T(n)$.

   (d) Show that your recurrence relation has as its solution $T(n) = 3n/2 - 2$.