EECS660 **HW1** **Due: 9/11/14**

1. (20) Using the definition of big-O and big-Ω to prove or disprove the following statements.

   $(i)\ \dfrac{n^3 - 8n^2 \lg n + 5n^2 - 10}{6n^2 - 7n^{3/2} + 3n - 8} = O(n).$

   $(ii)\ \dfrac{n^3 - 8n^2 \lg n + 5n^2 - 10}{6n^2 - 7n^{3/2} + 3n - 8} = \Omega(n).$

2. (20) Prove that $\lg n! = \Theta(n \lg n).$

3. (20) Prove or disprove the following statements:

   (a) $3^n = O(2^{n+2}).$

   (b) If $f(n) = O(g(n)),$ then $2^{f(n)} = O(2^{g(n)}).$

4. (10) Construct an example using two (eventually) positive functions $f(n)$ and $g(n)$ such that

   $f(n) = \Theta(g(n))$ but $\lim_{n\to\infty} \dfrac{f(n)}{g(n)}$ does not exist. Justify your answer.

5. (20) Given an array A[1..n] with n distinct integers and an integer key x. Assuming that $\Pr(x = A[i]) = \dfrac{1}{3^i}, \forall i,$ based on the number of comparisons between x and A[i]'s, compute $T_a(n)$ in closed-form if sequential search algorithm is used for searching x in A (starting at A[1]). You must set up the equation for $T_a(n)$ and then evaluate the sum.

6. (10) Given the following algorithm for finding the two largest integers in an array A[1..n] of n distinct positive integers. In terms of the number of comparisons between elements in A, compute $T_a(n)$. Justify your answer.

```
if A[1] > A[2]                    // Initialization
   then largest = A[1];
        s_largest = A[2]
   else  largest = A[2];
        s_largest = A[1]
endif;
for i = 3 to n do                 // Checking A[3], ..., A[n]
   if A[i] > s_largest            // A[i] is one of the two largest integers
      then if A[i] > largest      // A[i] is the current largest integer
              then    s_largest = largest;
                      largest = A[i]
              else    s_largest = A[i]
           endif
   endif
endfor;
```

Remark: Turn in Problems 2, 5, and 6 for grading.