1. The following pseudo code shows an implementation of the selection sort algorithm.

   1:  function Selection-Sort(A, n)
   2:    for i = 1 to n-1 do
   3:      min ← i
   4:      for j = i + 1 to n do
   6:          min ← j
   7:      end if
   8:    end for
   9:    swap A[i], A[min]
  10:   end for
  11:  end function

(a) Compute the worst case running time using the method shown in class for insertion sort. That is, assign a different constant to each of the lines 2-10 and use them to compute the running time.

(b) Repeat part (a) for the best case running time.

(c) Use the O-notation to compare the worst-case and best-case running times computed above to the following functions \( n \), \( n \lg n \), and \( n^2 \).

(d) Compare the worst and best case running times of the selection sort to the corresponding times of the insertion sort using one of the three notations, \( \Theta \), \( o \), or \( \omega \).

2. Use L'Hôpital's theorem to prove that:

\[
\log(n)^{k_1} = o(n^{k_2})
\]

For any values of \( k_1 \) and \( k_2 \) including the case where \( k_1 \) is not integer.

3. Rank the following functions by order of growth; that is, find an arrangement \( g_1, g_2, \ldots \) of the functions satisfying \( g_1 = \Omega(g_2), g_2 = \Omega(g_3), \ldots \). Partition your list into equivalence classes such that functions \( f(n) \) and \( g(n) \) are in the same class if and only if \( f(n) = \Theta(g(n)) \).

\[
\begin{align*}
& (\sqrt{2})^{\lg n} \quad n^2 \quad n! \quad (3/2)^n \quad n^3 \quad \lg^2 n \quad \lg(n!) \quad 2^{2^n} \quad \ln \ln n \quad 1 \quad \ln n \quad e^n \quad (n + 1)! \quad \sqrt{\lg n} \\
& \quad n \quad 2^n \quad n \lg n \quad 2^{2^n + 1}
\end{align*}
\]

Good luck!