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

```
function Selection-Sort(A, n)
    for i = 1 to n-1 do
        min ← i
        for j = i + 1 to n do
                min ← j
            end if
        end for
        swap A[i], A[min]
    end for
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. Is the following statement true or false?

\[ 2^n = \Theta(3^n) \]

Justify your answer using the basic definition of the \( \Theta \)-notation.

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 \( f(n) = \Theta(g(n)) \).

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