Problem Set 1

Assigned: May 27
Due: June 3

PROBLEM 1

For each of the following pairs of functions f (n) and g(n), state whether f is o(g), f is Θ(g); or g is o(f). (Exactly one of these is true in all cases.)

a. f (n) = n^{10}: g(n) = $2^{n/2}$.

   Ans: f is o(g)

b. f (n) = $n^{3/2}$; g(n) = n $\log^2(n)$.

   Ans: g is o(f)

c. f (n) = log(n^3); g(n) = log(n).

   Ans: f is Θ(g)

d. f (n) = log(3^N); g(n) = log(2^N).

   Ans: f is Θ(g)

e. f (n) = (log(n))^3; g(n) = (log(n))

   Ans: g is o(f)

f. f (n) = 2^N ; g(n) = $2^{N/2}$.

   Ans: f is Θ(g)

g. f (n) = n^2; g(n) = $(n/2)^2$.

   Ans: f is Θ(g)

h. f (n) = n^2; g(n) = $(n + 2)^2$

   Ans: f is Θ(g)

i. f (n) = 2^N ; g(n) = 2^{N+2}

   Ans: f is Θ(g)

j. f (n) = n!; g(n) = (n + 2)!

   Ans: f is Θ(g)

PROBLEM 2

The following two functions both compute the same thing. They take as arguments two arrays A and B and return TRUE if every element of A is less than every element of B.

```
AllLessThan1(int[] A,B) return bool{
 for (i=1 to|A|)
    for (j=1 to |B|)
      if (A[i] >= B[j]) return FALSE;
return TRUE;
}

AllLessThan2(int[] A,B) return bool{
 largeA = A[1]
  for (i = 2 to |A|)
    if (A[i]> largeA) largeA = A[i];
  for (j = 1 to |B|)
    if (largeA >= B[j]) return FALSE;
   return TRUE;
}
```

A. Give the asymptotic worst-case running time of each algorithm as a function of |A| and |B|. When does the worst case occur?

Ans: For AllLessThan1, the asymptotic worst-case running time would be O(|A|.|B|) and for AllLessThan2, it would be O(|A| + |B|).

This will occur when none of the elements in A are greater than those in B.

B. Give the asymptotic best-case running time of each algorithm as a function of |A| and |B|. When does the best case occur?

Ans: For AllLessThan1, the asymptotic best-case running time would be O(1) and this will occur when the first element of A is greater than or equal to the first element of B.
For AllLessThan2, it would be O(|A|) and this will occur when the first element of B is less than or equal to the greatest element in A.

C. Design an algorithm whose best-case running time is as good as the best-case for both of these algorithms, and whose worst-case running time is as good as the worst-case both of these algorithms.

Ans:

```
AllLessThan3(int[] A,B) return bool{
      LargeA = A[1]                  //LargeA contains the largest element
                                     //of A till the loop iteration
      SmallB = B[1]                  //SmallB contains the smallest element
                                     //of B till the loop iteration

      if (LargeA >= SmallB) return FALSE
      endif

      if(|A| < |B|)
            for(i = 2 to |B|)
                  if(i <= |A|)
                        if(LargeA < A[i]) LargeA = A[i]
                        endif
                  endif
                  if(SmallB > B[i]) SmallB = B[i]
                  endif
                  if(LargeA >= SmallB) return FALSE
                  endif
      else
            for(i = 2 to |A|)
                  if(LargeA < A[i]) LargeA = A[i]
                  endif
                  if(i <= |B|)
                        if(SmallB > B[i]) SmallB = B[i]
                        endif
                  endif
                  if(LargeA >= SmallB) return FALSE
                  endif
      endif
      return TRUE
}
```

The best case running time for the above mentioned algorithm would be O(1) and this would occur when the first element of A is greater than the first element of B.

The worst case running time would be either O(|B|) if |B| is greater than |A| else O(|A|) and this will occur when none of the elements in A are greater than those in B or even the greatest element in A is smaller than the smallest element in B.