Foundations of Algorithms
Homework #1

All members of the collaboration group are expected to participate fully in solving collaborative problems, and peers will assess performance at the end of the assignment. Note, however, that each student is required to write up their solutions individually. Common solution descriptions from a collaboration group will not be accepted. Furthermore, to receive credit for a collaboration problem, each student in the collaboration group must actively and substantially contribute to the collaboration. This implies that no single student should post a complete solution to any problem at the beginning of the collaboration process.

Problems for Grading

1. [20 points] Analyze the following algorithms for performing searches.

   (a) [10 points] Describe the complexity of the linear search algorithm.

   ```
   LINEAR-SEARCH($x, A$)
   i = 0
   while $i < n$ and $x \neq a_i$
       i = i + 1
   if $i < n$
       location $= i$
   else location $= nil$
   return location
   ```

   (b) [10 points] Describe the time complexity of the binary search algorithm in terms of number of comparisons used (ignore the time required to compute $m = \lfloor (i+j)/2 \rfloor$ in each iteration of the loop in the algorithm) as well as showing the worst case complexity. *Note*: Do not use the algorithm on page 799 of the book, use the following algorithm.

   ```
   BINARY-SEARCH($x, A$)
   i = 0
   j = n-1
   while $i < j$
       $m = \lfloor (i+j)/2 \rfloor$
       if $x > a_m$
           $i = m + 1$
       else $j = m$
   if $x = a_i$
       location $= i$
   else location $= -1$
   return location
   ```

2. [10 points] Give asymptotic upper and lower bounds for $T(n) = 3T(n/2) + n \lg n$, assuming $T(n)$ is constant for sufficiently small n. Make your bounds as tight as possible. Prove that your bounds are correct.

3. [15 points] Give asymptotic upper and lower bounds for $T(n) = T(\sqrt{n}) + 1$, assuming $T(n)$ is constant for sufficiently small n. Make your bounds as tight as possible. Prove that your bounds are correct.

4. [20 points] Give asymptotic upper and lower bounds for $T(n) = 2T\left(\frac{n}{3} + 1\right) + n$, assuming $T(n)$ is constant for sufficiently small n. Make your bounds as tight as possible. Prove that your bounds are correct.

5. [35 points] ***Collaborative Problem*** –CLRS 2-1: Although merge sort runs in $\Theta(n \lg n)$ worst-case time and insertion sort runs in $\Theta(n^2)$ worst-case time, the constant factors in insertion sort can make it faster in practice for small problem sizes on many machines. Thus, it makes sense to *coarsen* the leaves of the recursion by using insertion sort within merge sort when subproblems become sufficiently small. Consider a modification to merge sort in which n/k sublists of length k are sorted using insertion sort and then merged using the standard merging mechanism, where k is a value to be determined.

   (a) [10 points] Prove that insertion sort can sort the n/k sublists, each of length k, in $\Theta(nk)$ worst-case time.

   (b) [10 points] Prove how to merge the sublists in $\Theta(n \lg (n/k))$ worst-case time.

   (c) [10 points] Given that the modified algorithm runs in $\Theta(nk + n \lg (n/k))$ worst-case time, what is the largest value of k as a function of n for which the modified algorithm has the same running time as standard merge sort, in terms of Θ-notation?

   (d) [5 points] How should we choose k in practice?