CS3230
Tutorial 3

1. Draw a decision tree to show that 4 numbers can be sorted using at most 5 comparisons.

2. Given an array A of integers and a number k, give an efficient algorithm to decide if there exist two elements $A[i]$ and $A[j]$, $i \neq j$, such that $A[i] + A[j] = k$. Give time complexity bound of your algorithm.

3. Consider the following modification of the partition algorithm done in class. Show that it works correctly.

   Partition(A, i, j)
   Assumption: $i < j$
   1. Let $m = i + 1$, $n = j$;
   2. While $m \leq n$, do {
      3. While $A[m] < A[i]$ and $m \leq n$ do { $m = m + 1$ }
      4. While $A[n] \geq A[i]$ and $n \geq m$ do { $n = n - 1$ }
      5. If $n > m$, then swap($A[m]$,$A[n]$).
      }
   6. swap$(A[i], A[n])$
   7. Return n.

   End

4. Given as input a sorted array A, containing n elements, and two numbers ℓ and u (where $\ell \leq u$). Give an algorithm to find how many numbers are there in the array which are between ℓ and u (both inclusive). That is, find the number of x in the array A such that $\ell \leq x \leq u$.

   What is the time complexity of your algorithm.

5. Suppose we are given an array of n integers between 1 to m (both inclusive). Preprocess the array such that one can answer the following query in constant time:

   How many numbers are there in the array which are between ℓ and u (both inclusive), where $1 \leq \ell \leq u \leq m$.

   What is the time complexity of your preprocessing algorithm? Try to make it linear in m and n.