CS3230
Tutorial 2

1. Which of the following recurrence relations can be solved using the master theorem? If it cannot be solved, state so. If it can be solved, give the answer (in terms of Θ notation) and state which clause of the master theorem applies.

   (a) $T(n) = 15T(n/3) + n^3$.

   (b) $T(n) = 3T(n/2) + n$.

   (c) $T(n) = 4T(n/2) + n^2$.

   (d) $T(n) = T(n/3) + T(2n/3) + n$.

   (e) $T(n) = 4T(n/3) + n \log n$.

2. Use induction to detmine the upper bound for the following recurrence relations. Express your answer in big O-notation.

   (a) $T(n) = 2 + \sum_{i=1}^{n-1} T(i)$; $T(1) = 1$.

   (b) $T(n) = T(n-1) + 5n$; $T(1) = 5$.

   (c) $T(n) = T(n-1) + 3n^2$; $T(1) = 1$.

   (d) $T(n) = T(n-1) + \log n$; $T(1) = 1$.

   (e) $T(n) = 2T(n-1) + 1$; $T(1) = 1$.

3. Recall Euclid's algorithm for GCD. Assume $m \geq n$.

   Then,
   $$GCD(n,m) = \begin{cases} m, & \text{if } n = 0; \\ GCD(m \textbf{ mod } n, n), & \text{otherwise;} \end{cases}$$

   Give a good upper bound on the running time of the above algorithm in terms of m, n.

4. Binary Insertion Sort:

   Binary insertion sort is similar to insertion sort, except that: instead of finding the place to insert a new element using linear search, it does a binary search.

   Please give an algorithm to do binary insertion sort.

   Do a time-complexity analysis of your algorithm.

5. Prof Larry suggests to change the quicksort algorithm to:

```
Quicksort($A, i, j$)
    If $i < j$, then
        $p$ =Partition($A, i, j$)
        Quicksort($A, i, p$)
        Quicksort($A, p, j$)
End
```

Will the above algorithm work? If so, what is the complexity of the algorithm. If not, why not?

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