1. Let $f, g, h$ be any three positive integer functions.

Prove that if $f(n)$ is $O(h(n))$ and $g(n)$ is $O(h(n))$ then $f(n) + g(n)$ is $O(h(n))$. 

5 pts
2. Consider the recurrence relation:

\[ T(n) = \begin{cases} 
1 & \text{if } n = 1 \\
2 \cdot T(n/2) + 1 & \text{otherwise}
\end{cases} \]

Use the recursion tree method to solve the above recurrence relation. For simplicity, you may assume that \( n \) is a power of 2.

\[ 10 \text{ pts} \]
3. Give pseudocode for a function FIND-MAX that given an array $A$ of positive integers and an integer $i > 0$ returns the index $j$, $1 \leq j \leq i$, such that $A[j]$ is the maximum element in $A[1..i]$. Analyse the running time of your function.

$\textit{function} \ \text{FIND-MAX} \ (\text{int} \ A[], \ \text{int} \ i) \\
\triangleright \text{returns the index } j, \ 1 \leq j \leq i, \text{ such that } A[j] \text{ is} \\
\triangleright \text{the largest element in } A[1..i] \\
10 \ \text{pts}$
4. To sort an array $A[1..n]$ of positive integers, one can use the following recursive idea: Find the position $j$ of the maximum element in $A[1..n]$, swap $A[j]$ with $A[n]$ and then sort $A[1..n-1]$. Use this idea together with the function FIND-MAX from problem 3 to give pseudocode for a recursive sorting algorithm, REC-SORT.

15 pts
5. Let $T(n)$ denote the worst-case running time of REC-SORT on an array of size $n$.

(i) Derive a recurrence relation for $T(n)$.  

(ii) Solve the recurrence relation in part(i) to obtain the exact asymptotic rate of growth of $T(n)$.  

5 pts