In all cases explain clearly and as succinctly as possible.

1. (20 pts) Prove or disprove.
   (a) \( n^{1.001} + n \log n = \Theta(n^{1.001}) \).
   (b) \( 2^{\sqrt{n \log n}} = O(\sqrt{n}) \).
   (c) \( 5n^32^n = o(2^{1.1n}) \).
   (d) \( n^3/\log n = \Omega(n^2) \).
   (e) \( (\log n)^n = \omega(n^\log n) \).

2. (10 pts) Prove that \( \sum_{i=1}^{n} i^d \log(i) \) is in \( \Theta(n^{d+1} \log(n)) \).  \{ Hint: Use \( \int_{a}^{b} x^d \log(x) \, dx \) for appropriate values of \( a, b \) to bound the summation from above and below. \}

3. Assume that the elements of an Array \( A[] \) (of size 10) are:
   (a) (10 pts) Sort the array in increasing order using Insertion Sort.
   (b) (20 pts) Sort the array in increasing order using Quick Sort.
   (c) (20 pts) Sort the array in increasing order using Merge Sort.

   In cases where the array is sorted recursively, show the input to and the output from each recursive call in the recursion stack.

   Within each recursive call (and the case where the procedure is not recursive) show your steps.

4. (10 pts) Consider the following Pseudo code for Binary Search.

\begin{verbatim}
BinarySearch(A[],x,i,j) \ \// Find x in sorted A[] between indices i,j 
{
    if(i>j)
        return(not found);
    else
    {
        mid=(i+j) div 2;
        if(A[mid]==x)
            return(mid)
        else if(A[mid]>x)
            BinarySearch(A[],x,i,mid-1);
        else if(A[mid]<x)
            BinarySearch(A[],x,i,mid+1);
    }
}\end{verbatim}
Write a recurrence relation for the time complexity for BinarySearch and solve it.

5. (20 pts) Solve the following recurrence relations (without using the master theorem). In each case assume $T(n)$ in the base case of $n = 1$ to be something that simplifies your computation.

(a) $T(n) = 3T(n/2) + n$
(b) $T(n) = 2T(n/2) + n^3$
(c) $T(n) = 2T(n/4) + \sqrt{n}$
(d) $T(n) = T(\sqrt{n}) + 1$

6. (20 pts) Give a $\Theta(n \log n)$ time algorithm that, given an array $A[]$ of $n$ integers and another integer $x$, determines whether or not there exist two elements in $A[]$ whose sum is exactly $x$.

7. (20 pts) Let $A[1...n]$ be an array of $n$ distinct numbers. If $i < j$ and $A[i] > A[j]$ then the pair $(i, j)$ is called an inversion of $A[]$.

Now, suppose in $A[1...n]$ $(i, j)$ is an inversion for some value of $i, j$. If we swap $A[i]$ and $A[j]$, this removes at least one inversion in $A[]$. However, it may remove other inversions too. What is the maximum number of inversions (including the one corresponding to $(i, j)$) that the swap may remove?