Homework 1

Problem 1. Let $f(n)$ and $g(n)$ be positive functions. Prove by the definition of Θ that $F(n) = \Theta(G(n))$ where $F(n) = \max(f(n), g(n))$ and $G(n) = f(n) + g(n)$.

Problem 2. Show that for any real constants a and b, where $b > 0$

$$(n + a)^b = \Theta(n^b).$$

Problem 3. Find the running time of the following program. Give an asymptotic analysis of the running time using big-Oh (or big-Theta which would be technically more precise).

```
a=n;
while(a>1) {
   b=a;
   while(b<n) {
      for (c=0; c<n; c+=3)
         print "hello";
      b*=2;
   }
   a/=2;
}
```

Problem 4. Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, \ldots$ of the functions satisfying $g_1 = \Omega(g_2), g_2 = \Omega(g_3), \ldots$ Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n)$.

$(\sqrt{2})^{\lg n}$	n^2	$n!$	$\ln n$	$(\frac{3}{2})^n$	n^3
$\lg^2 n$	$\lg(n!)$	2^{2^n}	$n \lg n$	$\lg \lg n$	$n \cdot 2^n$
$4^{\lg n}$	$(n+1)!$	n	2^n	$2^{\lg n}$	e^n

Problem 5. Let $f(n)$ and $g(n)$ be positive functions. Prove or disprove each of the following conjectures.

a. $f(n) = O(g(n))$ implies $g(n) = O(f(n))$

b. $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$.