Problem-Set: Asymptotic Notations

Note: Read Chapter 3 (Growth of functions) of CLRS book "Introduction to Algorithms 3rd Edition"

P – 1 [30 Marks]

1. Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.
2. Show that for any real constants a and b, where $b > 0$, $(n+a)^b = \Theta(n^b)$
3. Explain why the statement, "The running time of algorithm A is at least $O(n^2)$," is meaningless.
4. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?
5. Prove that $o(g(n)) \cap \omega(g(n))$ is the empty set.
6. Prove that $n! = \omega(2^n)$ and $n! = o(n^n)$.

P – 2 [30 Marks]

Indicate and **Prove**, for each pair of expressions (A, B) in the table below, whether A is O, o, Ω, ω, or Θ of B. Assume that $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of the table with "yes" or "no" written in each box. Prove your answer for each case independently.

	A	***B***	O	o	Ω	ω	Θ
a.	$\lg^k n$	n^ϵ					
b.	n^k	c^n					
c.	$\sqrt{n}$	$n^{\sin n}$					
d.	2^n	$2^{n/2}$					
e.	$n^{\lg c}$	$c^{\lg n}$					
f.	$\lg(n!)$	$\lg(n^n)$					

P – 3 [30 Marks]

Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, \ldots, g_{30}$ of the functions satisfying $g_1 = \Omega(g_2)$, $g_2 = \Omega(g_3)$, ..., $g_{29} = \Omega(g_{30})$. Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$. **Justify/Prove your ranking**.

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

P – 4 [30 Marks]

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

a. $f(n) = O(g(n))$ implies $g(n) = O(f(n))$.
b. $f(n) + g(n) = \Theta(\min(f(n), g(n)))$.
c. $f(n) = O(g(n))$ implies $\lg(f(n)) = O(\lg(g(n)))$, where $\lg(g(n)) \geq 1$ and $f(n) \geq 1$ for all sufficiently large n.
d. $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$.
e. $f(n) = O((f(n))^2)$.
f. $f(n) = O(g(n))$ implies $g(n) = \Omega(f(n))$.
g. $f(n) = \Theta(f(n/2))$.
h. $f(n) + o(f(n)) = \Theta(f(n))$.

P – 5 [30 Marks]

The iteration operator* used in the lg* function can be applied to any monotonically increasing function $f(n)$ over the reals. For a given constant $c \in \mathbf{R}$, we define the iterated function f_c^* by $f_c^*(n) = \min\{i \geq 0 : f^{(i)}(n) \leq c\}$, which need not be well-defined in all cases. In other words, the quantity $f_c^*(n)$ is the number of iterated applications of the function f required to reduce its argument down to c or less. For each of the following functions $f(n)$ and constants c, give as tight a bound as possible on $f_c^*(n)$.

	***f*(*n*)**	***c***	$f_c^*(n)$
a.	$n - 1$	0	
b.	$\lg n$	1	
c.	$n/2$	1	
d.	$n/2$	2	
e.	$\sqrt{n}$	2	
f.	$\sqrt{n}$	1	
g.	$n^{1/3}$	2	
h.	$n/\lg n$	2	

P – 6 [30 Marks]

In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both (in which case $f = \Theta(g)$).

	$f(n)$	$g(n)$
(a)	$n - 100$	$n - 200$
(b)	$n^{1/2}$	$n^{2/3}$
(c)	$100n + \log n$	$n + (\log n)^2$
(d)	$n \log n$	$10n \log 10n$
(e)	$\log 2n$	$\log 3n$
(f)	$10 \log n$	$\log(n^2)$
(g)	$n^{1.01}$	$n \log^2 n$
(h)	$n^2/\log n$	$n(\log n)^2$
(i)	$n^{0.1}$	$(\log n)^{10}$
(j)	$(\log n)^{\log n}$	$n/\log n$
(k)	$\sqrt{n}$	$(\log n)^3$
(l)	$n^{1/2}$	$5^{\log_2 n}$
(m)	$n2^n$	3^n
(n)	2^n	2^{n+1}
(o)	$n!$	2^n
(p)	$(\log n)^{\log n}$	$2^{(\log_2 n)^2}$
(q)	$\sum_{i=1}^{n} i^k$	n^{k+1}