Homework0916 Due date: 20210923, 1 P.M.

1-1 *Relative asymptotic growths*

Indicate, 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.

	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.	$\lg(n!)$	$\lg(n^n)$					

1-1 *Relative asymptotic growths*

Indicate, 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.

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

1-2 *Ordering by asymptotic growth rates*

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

$\lg(\lg^* n)$	$2^{\lg^* n}$	$(\sqrt{2})^{\lg n}$	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	$n \lg n$	$2^{2^{n+1}}$

$$\lg^* n = \min\left\{i \geq 0 : \lg^{(i)} n \leq 1\right\} .$$

The iterated logarithm is a *very* slowly growing function:

$$\begin{aligned}
\lg^* 2 &= 1 , \\
\lg^* 4 &= 2 , \\
\lg^* 16 &= 3 , \\
\lg^* 65536 &= 4 , \\
\lg^* (2^{65536}) &= 5 .
\end{aligned}$$

$$2^{2^{n+1}}$$
$$2^{2^n}$$
$$(n+1)!$$
$$n!$$
$$e^n$$
$$n \cdot 2^n$$
$$2^n$$
$$(3/2)^n$$
$$(\lg n)^{\lg n} = n^{\lg \lg n}$$
$$(\lg n)!$$

$$n^3$$
$$n^2 = 4^{\lg n}$$
$$n \lg n \text{ and } \lg(n!)$$
$$n = 2^{\lg n}$$
$$(\sqrt{2})^{\lg n} (= \sqrt{n})$$
$$2^{\sqrt{2 \lg n}}$$

$$\lg^2 n$$
$$\ln n$$
$$\sqrt{\lg n}$$
$$\ln \ln n$$
$$2^{\lg^* n}$$
$$\lg^* n \text{ and } \lg^*(\lg n)$$
$$\lg(\lg^*) n$$
$$n^{1/\lg n} (= 2) \text{ and } 1$$

1-3 *Asymptotic notation properties*

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 $2^{f(n)} = O\left(2^{g(n)}\right)$.

Let $f(n)$ and $g(n)$ be asymptotically positive functions.

a. No, $f(n) = O(g(n))$ does not imply $g(n) = O(f(n))$. Clearly, $n = O(n^2)$ but $n^2 \neq O(n)$.

b. No, $f(n) + g(n)$ is not $\Theta(\min(f(n), g(n)))$. As an example notice that $n + 1 \neq \Theta(\min(n, 1)) = \Theta(1)$.

c. Yes, if $f(n) = O(g(n))$ then $\lg(f(n)) = O(\lg(g(n)))$ provided that $f(n) \geqslant 1$ and $\lg(g(n)) \geqslant 1$ are greater than or equal 1. We have that:

$$f(n) \leqslant cg(n) \Rightarrow \lg f(n) \leqslant \lg(cg(n)) = \lg c + \lg g(n)$$

To show that this is smaller than $b \lg g(n)$ for some constant b we set $\lg c + \lg g(n) = b \lg g(n)$. Dividing by $\lg g(n)$ yields:

$$b = \frac{\lg(c) + \lg g(n)}{\lg g(n)} = \frac{\lg c}{\lg g(n)} + 1 \leqslant \lg c + 1$$

The last inequality holds since $\lg g(n) \geqslant 1$.

d. No, $f(n) = O(g(n))$ does not imply $2^{f(n)} = O(2^{g(n)})$. If $f(n) = 2n$ and $g(n) = n$ we have that $2n \leqslant 2 \cdot n$ but not $2^{2n} \leqslant c2^n$ for any constant c by exercise $3.1 - 4$.

1-4 Consider sorting n numbers stored in array A by first finding the largest element of A and exchanging it with the element in A[n]. Then find the second largest element of A, and exchange it with A[n-1]. Continue in this manner for all n elements of A. Write pseudocode for this algorithm, and answer the following questions: What loop invariant does this algorithm maintain? Give the best-case and worst-case running times of selection sort in asymptotic notation.

Answer:

Loop invariant: At the start of the i-th iteration, the subarray A[n-i+1,n] consists of the largest i elements of A in sorted order.

Best-case running time: $\Theta(n^2)$

Worst-case running time: $\Theta(n^2)$