5/9/93 Alex Campbell PROBLEM SET 1 CSCI 3104 SPRING 2014

1) a) $n+3 \overset{f(n)}{=} O(n^3) = g(n)$

$$\frac{f(n)}{g(n)} = \frac{\cancel{n}+3}{n^3} \rightarrow \frac{3}{n^2} \quad \lim_{n\to\infty} \frac{3}{n^2} = 0 \;\therefore\; n^3 \text{ is loose upper bound for } n+3 \rightarrow \text{TRUE}$$

b) $f(n) = 3^{2n} = O(3^n) = g(n) \rightarrow$ FALSE

$$\frac{f(n)}{g(n)} = \frac{3^{2n}}{3^n} \xrightarrow{LH} \frac{9^n \log(9)}{3^n \log(3)} \rightarrow \frac{9^n c}{3^n c} \quad \lim_{n\to\infty} \frac{9^n}{3^n} = \infty$$

$\therefore\; 3^n = O(3^{2n})$

c) $n^n = o(n!) \rightarrow$ FALSE

$$n! = n(n-1)(n-2)(n-3)\ldots \oplus n^n \quad \lim_{n\to\infty} \frac{n^n}{n!} = \infty$$

$\therefore\; n! = o(n^n)$

d) $f(n) = \frac{1}{3n} = o(1) = g(n)$

$$\frac{f(n)}{g(n)} = \frac{\frac{1}{3n}}{1} \quad \lim_{n\to\infty} \frac{\frac{1}{3n}}{1} = 0 \;\therefore\; 1 \text{ is upper bound for } \tfrac{1}{3n}$$

$\rightarrow$ TRUE

e) $\ln^3 n = \Theta(\lg^3 n)$

$$\frac{\ln_e^3 (n)}{\lg_2^3 (n)} \rightarrow \left(\frac{\lg_2 n}{\lg_2 e}\right)^3 = \Theta(\lg^3 n) \rightarrow c_1 \le \left(\frac{1}{\lg_2 e}\right)^3 \le c_2$$

$\hookrightarrow c_1(\cancel{\lg^3} n) \le \left(\frac{\cancel{\lg n}}{\lg_2 e}\right)^3 \le c_2(\cancel{\lg^3} n)$

$\hookrightarrow$ constant

$\therefore$ TRUE

2) a) $\frac{d}{dt}\left(3t^4 + \frac{1}{3}t^3 - 7\right) = \boxed{12t^3 + t^2}$

b) $\sum_{i=0}^{k} 2^i$

$2^3 = 8$

$k=0 \rightarrow 1 +$
$k=1 \rightarrow 2 +$
$k=2 \rightarrow 4 +$
$k=3 \rightarrow 8 +$

$= 15 \rightarrow \boxed{2^{k+1} - 1}$

c) $\Theta\left(\sum_{k=1}^{n} 1/k\right)$

$$\Theta\left(\int_1^n 1/k\right) \rightarrow \Theta\left(\ln x \Big|_1^n\right) \rightarrow \Theta\left(\ln(n) - \ln(1)\right) \rightarrow \boxed{\Theta(\ln(n))}$$

3) In order (Symmetric) tree traversal is an $O(n)$ time algorithm that will work. This algorithm works by traversing to the left-most node (lowest int), followed by its root, followed by the roots right node. Therefore ascending order is achieved. This is order n because for every node added, time it takes to complete increases linearly.

4)

a) $1.99^{41} = 20.72$ days $> 41^3 = 7.97 \times 10^{-7}$ days $+ 17$ days $= x < 20.72$

$\therefore$ we want to create the new algorithm

b) $(10^6)^2 = 11.57$ days $< (10^6)^{1.99} = 10.08$ days $+ 2$ days $= 12.08$ days

$\therefore$ we want to use the existing algorithm

5) a) $2^{nk} = O(2^n)$ for $k > 1$?

$$\lim_{n \to \infty} \frac{2^{nk}}{2^n} = \frac{2^{nk-n}}{1} = \infty \therefore 2^{nk} \text{ grows faster than } 2^n \rightarrow \text{FALSE}$$

b) $2^{n+k} = O(2^n)$ for $k = O(1)$

$$\frac{2^{n+k}}{2^n} = \frac{2^k}{1} \rightarrow \text{Constant} \rightarrow 2^{n+k} = \Theta(2^n) \therefore \text{if } \Theta \text{ is true then } O \text{ must be true.} \rightarrow \text{TRUE}$$

6) Yes. In a min-heap tree, the parent $\leq$ the children, OR children $\geq$ parent. With this in mind, our tree indices will be in ascending order, similarly to our sorted order array.