1. For each of the following functions, indicate the class (g(n)) the function belongs to. (Use the simplest g(n) possible in your answers.) Prove your assertions.

a. $(n^2 + 10)^{10}$

As $n \to \infty$, the function $(n^2 + 10)^{10} \approx (n^2)^{10} = n^{20}$

Therefore, $(n^2 + 10)^{10} \in \Theta(n^{20})$

<u>Poof:</u> *Let* $f(n) = (n^2 + 10)^{10}$, $g(n) = n^{20}$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{(n^2+10)^{10}}{n^{20}}$$

$$= \lim_{n\to\infty} (\frac{n^2+10}{n^2})^{10}$$

$$= \lim_{n\to\infty} (\frac{n^2}{n^2} + \frac{10}{n^2})^{10}$$

$$= \lim_{n\to\infty} (1 + \frac{10}{n^2})^{10}$$

$$= (1 + \frac{10}{\infty})^{10} = (1 + 0)^{10} = (1)^{10} = 1$$ (contant)

Since it evaluated to a constant value, the function $(n^2 + 10)^{10}$ is in the efficiency class $\Theta(n^{20})$

b. $\sqrt{10n^2 + 7n + 3}$

As $n \to \infty$, the function $\sqrt{10n^2 + 7n + 3} \approx \sqrt{n^2 + n} = \sqrt{n(n + 1)} \approx n$

Therefore, $\sqrt{10n^2 + 7n + 3} \in \Theta(n)$

<u>Poof:</u> $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ *Let* $f(n) = \sqrt{10n^2 + 7n + 3}$, $g(n) = n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\sqrt{10n^2+7n+3}}{n}$$

$$= \lim_{n\to\infty} (\frac{10n^2+7n+3}{n^2})^{1/2}$$ (substitute n with $\sqrt{n^2}$)

$$= \lim_{n\to\infty} (\frac{10n^2}{n^2} + \frac{7n}{n^2} + \frac{3}{n^2})^{1/2}$$

$$= \lim_{n\to\infty} (10 + \frac{7}{n} + \frac{3}{n^2})^{1/2}$$

$$= (10 + \frac{7}{\infty} + \frac{3}{\infty^2})^{1/2} = (10 + 0 + 0)^{1/2} = (10)^{1/2} \approx 3.16$$ (constant)

Since it evaluated to a constant value, the function $\sqrt{10n^2 + 7n + 3}$ is in the efficiency class $\Theta(n)$

c. $2n \lg(n + 2)^2 + (n + 2)^2 \lg(\frac{n}{2})$

As $n \to \infty$, the function $[2n \lg(n + 2)^2 + (n + 2)^2 \lg(\frac{n}{2})] \approx n \lg(n^2) + n^2 \lg n$

Therefore, $[2n \lg(n + 2)^2 + (n + 2)^2 \lg(\frac{n}{2})] \in \Theta(n \log n) + \Theta(n^2 \log n) = \Theta(max\{n \log n, n^2 \log n\})$

More simply, $[2n\lg(n+2)^2 + (n+2)^2\lg(\frac{n}{2})] \in \Theta(n^2\log n)$

First prove $2n\lg(n+2)^2 \in \Theta(n\log n)$ then prove $(n+2)^2\lg(\frac{n}{2}) \in \Theta(n^2\log n)$

*NOTE: Going to use '$\lg$' as the notation instead of '$\log$' to make things easier to follow. Also because Prof. Wang said that $\lg = \log_{10}$ (same as '$\log$' so it shouldn't make a difference).

<u>Proof #1:</u> *Let* $F(n) = 2n\lg(n+2)^2$, $G(n) = n\lg n$

$$\lim_{n\to\infty} \frac{F(n)}{G(n)} = \lim_{n\to\infty} \frac{2n\lg(n+2)^2}{n\lg n}$$

$$= \lim_{n\to\infty} \frac{4n\lg(n+2)}{n\lg n}$$

$$= \lim_{n\to\infty} \frac{4\lg(n+2)}{\lg n} \qquad \text{(n cancel out)}$$

$$= \lim_{n\to\infty} \frac{(4\lg(n+2))'}{(\lg n)'} = \lim_{n\to\infty} [\frac{4}{(n+2)\ln 10}]/[\frac{1}{n\ln 10}] \qquad \text{(Use L'Hopital's rule)}$$

$$= \lim_{n\to\infty} [\frac{4}{(n+2)\ln 10}] \cdot [\frac{n\ln 10}{1}]$$

$$= \lim_{n\to\infty} \frac{4n\ln 10}{(n+2)\ln 10}$$

$$= \lim_{n\to\infty} \frac{4n}{n+2}$$

$$= \lim_{n\to\infty} \frac{(4n/n)}{(n/n)+(2/n)} \qquad \text{(divide everything by n)}$$

$$= \lim_{n\to\infty} \frac{4}{1+(2/n)}$$

$$= \frac{4}{1+(2/\infty)} = \frac{4}{1+0} = \frac{4}{1} = 4 \qquad \text{(constant)}$$

<u>Proof #2:</u> *Let* $f(n) = (n+2)^2\lg(\frac{n}{2})$, $g(n) = n^2\lg n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{(n+2)^2\lg(n/2)}{n^2\lg n}$$

$$= \lim_{n\to\infty} \frac{(n+2)^2(\lg n - \lg 2)}{n^2\lg n}$$

$$= \lim_{n\to\infty} (n+2)^2(\frac{\lg n}{n^2\lg n} - \frac{\lg 2}{n^2\lg n}) \qquad \text{(expand whole equation)}$$

$$= \lim_{n\to\infty} (n+2)[\frac{1}{n} - \frac{\lg 2}{n\lg n} + \frac{2}{n^2} - \frac{\lg 4}{n^2\lg n}]$$

$$= \lim_{n\to\infty} (1 - \frac{\lg 2}{\lg n} + \frac{2}{n} - \frac{\lg 4}{n\lg n} + \frac{2}{n} - \frac{\lg 4}{n\lg n} + \frac{4}{n^2} - \frac{\lg 16}{n^2\lg n})$$

$$= \lim_{n\to\infty} (1 - \frac{\lg 2}{\lg n} + \frac{4}{n} - \frac{\lg 16}{n\lg n} + \frac{4}{n^2} - \frac{\lg 16}{n^2\lg n}) \qquad \text{(combine like terms)}$$

$$= 1 - \frac{\lg 2}{\infty} + \frac{4}{\infty} - \frac{\lg 16}{\infty} + \frac{4}{\infty} - \frac{\lg 16}{\infty}$$

$$= 1 - 0 + 0 - 0 + 0 - 0 \quad = 1 \qquad \text{(constant)}$$

Since both function evaluated to constants, the combined function $[2n\lg(n+2)^2 + (n+2)^2\lg(\frac{n}{2})]$ **is in the efficiency class** $\Theta(max\{n\log n, n^2\log n\})$ **or more simply** $\Theta(n^2\log n)$**.**

d. $2^{n+1} + 3^{n-1}$

As $n \to \infty$, the function $2^{n+1} + 3^{n-1} \approx 2^n + 3^n$

Therefore $(2^{n+1} + 3^{n-1}) \in \Theta(2^n) + \Theta(3^n) = \Theta(max\{2^n, 3^n\})$

Simplified to $(2^{n+1} + 3^{n-1}) \in \Theta(3^n)$

Proof: *Let* $f(n) = 2^{n+1} + 3^{n-1}$, $g(n) = 3^n$

$$\lim_{n\to\infty}\frac{f(n)}{g(n)} = \lim_{n\to\infty}\frac{2^{n+1}+3^{n-1}}{3^n}$$

$$= \lim_{n\to\infty}\frac{2^n2+3^n3^{-1}}{3^n}$$

$$= \lim_{n\to\infty}\frac{2\cdot2^n}{3^n} + \frac{3^n3^{-1}}{3^n}$$

$$= \lim_{n\to\infty} 2(\frac{2^n}{3^n}) + \frac{1}{3}(\frac{3^n}{3^n})$$

$$= \lim_{n\to\infty} 2(2^n/2^{n\cdot\log_2 3}) + \frac{1}{3}(1)$$ (rewrite 3^n to have base 2)

$$= \lim_{n\to\infty} 2(2^{n-n\cdot\log_2 3}) + \frac{1}{3}$$

$$= \lim_{n\to\infty} 2(2^{n(1-\log_2 3)}) + \frac{1}{3}$$

$$= \lim_{n\to\infty} 2(2^{n\cdot\log_2(2/3)}) + \frac{1}{3}$$ (apply log rules to $1 - \log_2 3$)

$$= \lim_{n\to\infty} 2(2^{\log_2(2/3)})^n + \frac{1}{3}$$ (bring n power outside bracket)

$$= \lim_{n\to\infty} 2(\frac{2}{3})^n + \frac{1}{3}$$

$$= 2(\frac{2}{3})^\infty + \frac{1}{3} = 2(0) + \frac{1}{3} = \frac{1}{3}$$ (constant)

Since the result is a constant value, the function $2^{n+1} + 3^{n-1}$ **is in the efficiency class** $\Theta(3^n)$**.**

e. $\lfloor\log_2 n\rfloor$

Can rewrite it as $\log_2 x - \epsilon$ where $0 < \epsilon < 1$. So as $n \to \infty$, the function $\lfloor\log_2 n\rfloor \approx \log_2 n$, therefore the function $\lfloor\log_2 n\rfloor \in \Theta(\log n)$

Proof: *Let* $f(n) = \log_2 n - \epsilon$, $g(n) = \log n$

$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\log_2 n - \epsilon}{\log n}$

$= \lim_{n\to\infty} \frac{\log_2 n - \epsilon}{\log n}$

$= \lim_{n\to\infty} \frac{(\log_2 n - \epsilon)'}{(\log n)'} = \lim_{n\to\infty} [\frac{1}{n\ln 2}]/[\frac{1}{n\ln 10}]$ (use L'Hopital's rule)

$= \lim_{n\to\infty} [\frac{1}{n\ln 2}] \cdot [\frac{n\ln 10}{1}]$

$= \lim_{n\to\infty} \frac{n\ln 10}{n\ln 2}$

$= \lim_{n\to\infty} \frac{\ln 10}{\ln 2} = \log_2 10$ (constant)

Since the result is a constant value, the function $\lfloor \log_2 n \rfloor$ is in the efficiency class $\Theta(\log n)$.

---

2. List the following functions according to their order of growth from the lowest to the highest: $(n-2)!$, $5\lg(n+100)^{10}$, 2^{2n}, $0.001n^4 + 3n^3 + 1$, $\ln^2 n$, $\sqrt[3]{n}$, 3^n

Determine efficiency classes:

- $(n-2)! \approx n!$ therefore $(n-2)! \in \Theta(n!)$
- $5\lg(n+100)^{10} = 50\lg(n+100) \approx 50\lg n$ therefore $50\lg n \in \Theta(\log n)$
- $2^{2n} = (2^2)^n = 4^n$ therefore $2^{2n} \in \Theta(4^n)$
- $0.001n^4 + 3n^3 + 1 \approx 0.001n^4 + 3n^3$ therefore $(0.001n^4 + 3n^3 + 1) \in \Theta(n^4) + \Theta(n^3) = \Theta(max\{n^4, n^3\}) = \Theta(n^4)$
- $\ln^2 n \in \Theta(\log^2 n)$
- $\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$
- $3^n \in \Theta(3^n)$

Roughly order by efficiency classes:

$\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$

$\ln^2 n \in \Theta(\log^2 n)$

$5\lg(n+100)^{10} \in \Theta(\log n)$

$(0.001n^4 + 3n^3 + 1) \in \Theta(n^4)$

$3^n \in \Theta(3^n)$

$2^{2n} \in \Theta(4^n)$

$(n-2)! \in \Theta(n!)$

Check $\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$ against $\ln^2 n \in \Theta(\log^2 n)$:

$\lim_{n\to\infty} \frac{(\sqrt[3]{n})'}{(\log^2 n)'} = \lim_{n\to\infty} [\frac{1}{3(n)^{4/3}}] \div [\frac{2\log n}{n\ln b}]$ (where b is a random base > 1)

$= \lim_{n \to \infty} [\frac{1}{3(n)^{4/3}}] \cdot [\frac{n\ln b}{2\log n}] = \lim_{n \to \infty} \frac{n\ln b}{3(n)^{2/3} 2\log n}$

$= \lim_{n \to \infty} \frac{(\sqrt[3]{n}\ln b)'}{(6\log n)'} = \lim_{n \to \infty} [\frac{1\ln b}{3(n)^{2/3}}] \div [\frac{6}{n\ln b}]$ (use L'Hopital's rule)

$= \lim_{n \to \infty} [\frac{1\ln b}{3(n)^{2/3}}] \cdot [\frac{n\ln b}{6}]$

$= \lim_{n \to \infty} \frac{n\ln^2 b}{18(n)^{2/3}} = \lim_{n \to \infty} \frac{\sqrt[3]{n}\ln^2 b}{18} = \frac{\sqrt[3]{\infty}\ln^2 b}{18} = \infty$

- Since it resulted in ∞ the function $\sqrt[3]{n}$ will grow faster than $\ln^2 n$

<u>Check</u> $\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$ against $5\lg(n + 100)^{10} \in \Theta(\log n)$:

$\lim_{n \to \infty} \frac{(\sqrt[3]{n})'}{(\log n)'} = \lim_{n \to \infty} \frac{1}{3(n)^{2/3}} \div \frac{1}{n\ln b}$

$\lim_{n \to \infty} \frac{1}{3(n)^{2/3}} \cdot \frac{1}{n\ln b} = \lim_{n \to \infty} \frac{n\ln b}{3(n)^{2/3}} = \lim_{n \to \infty} \frac{1}{3}\sqrt[3]{n}\ln b = \frac{1}{3}\infty \ln b = \infty$

- Since it resulted in ∞ the function $\sqrt[3]{n}$ will grow faster than $\log n$

<u>Check</u> $\ln^2 n \in \Theta(\log^2 n)$ against $5\lg(n + 100)^{10} \in \Theta(\log n)$:

$\lim_{n \to \infty} \frac{\log^2 n}{\log n} = \lim_{n \to \infty} \log n = \infty$

- Since it resulted in ∞, $\ln^2 n \in \Theta(\log^2 n)$ will grow faster than $5\lg(n + 100)^{10} \in \Theta(\log n)$.

<u>Check</u> $\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$ against $(0.001n^4 + 3n^3 + 1) \in \Theta(n^4)$:

$\lim_{n \to \infty} \frac{(\sqrt[3]{n})}{n^4} = \lim_{n \to \infty} \frac{1}{n^{11/3}} = \frac{1}{\infty} = 0$

- Since it resulted in 0, the function $0.001n^4 + 3n^3 + 1$ will grow faster than $\sqrt[3]{n}$.

<u>Check</u> $(0.001n^4 + 3n^3 + 1) \in \Theta(n^4)$ against $3^n \in \Theta(3^n)$:

$\lim_{n \to \infty} \frac{(n^4)'}{(3^n)'} = \lim_{n \to \infty} \frac{4n^3}{3^n\ln 3}$

$= \lim_{n \to \infty} \frac{(4n^3)'}{(3^n\ln 3)'} = \lim_{n \to \infty} \frac{12n^2}{3^n\ln^2 3}$

$= \lim_{n \to \infty} \frac{(12n^2)'}{(3^n\ln^2 3)'} = \lim_{n \to \infty} \frac{24n}{3^n\ln^3 3}$

$= \lim_{n \to \infty} \frac{(24n)'}{(3^n\ln^3 3)'} = \lim_{n \to \infty} \frac{24}{3^n\ln^4 3} = \frac{24}{\infty \cdot \ln^4 3} = 0$

- Since it resulted in 0, the function 3^n will grow faster than $0.001n^4 + 3n^3 + 1$.

<u>Check</u> $3^n \in \Theta(3^n)$ against $2^{2n} \in \Theta(4^n)$:

- Since $2^{2n} = 4^n$ we can see that for any given $n > 0$, $4^n > 3^n$ because their bases are different, therefore $2^{2n} \in \Theta(4^n)$ will grow faster than $3^n \in \Theta(3^n)$.

<u>Check</u> $2^{2n} \in \Theta(4^n)$ against $(n - 2)! \in \Theta(n!)$:

$\lim_{n\to\infty} \frac{4^n}{n!} \approx \lim_{n\to\infty} \frac{4^n}{\sqrt{2\pi n}(\frac{n}{e})^n}$ (using Sterling's formula)

$= \lim_{n\to\infty} [4^n]/[(\sqrt{2\pi n})(n^n)(\frac{1}{e})^n]$

$= \lim_{n\to\infty} [(4)/((2\pi n)^{(1/2n)} \frac{n}{e})]^n$

$= [(4)/(1 \cdot \infty)]^\infty$ ($a^{(1/\infty)} = 1$ where a is anything)

$= (0)^\infty = 0$

- Since it resulted in 0, the function $(n - 2)!$ will grow faster than 2^{2n}.

The order of growth for the functions (in ascending order) is:

1) $5\lg(n + 100)^{10} \in \Theta(\log n)$
2) $\ln^2 n \in \Theta(\log^2 n)$
3) $\sqrt[3]{n} \in \Theta(\sqrt[3]{n})$
4) $(0.001n^4 + 3n^3 + 1) \in \Theta(n^4)$
5) $3^n \in \Theta(3^n)$
6) $2^{2n} \in \Theta(4^n)$
7) $(n - 2)! \in \Theta(n!)$

---

3. Compute the following sums.

a. $1 + 3 + 5 + 7 + \cdots + 999$

It's an arithmetic sequence that follows the pattern $2i - 1$

$= \Sigma_{i=1}^{500}(2i - 1)$

$= \Sigma_{i=1}^{500}(2i) - \Sigma_{i=1}^{500}(1)$

$= 2 \cdot \Sigma_{i=1}^{500}(i) - \Sigma_{i=1}^{500}(1)$

$= 2 \cdot \frac{500(500+1)}{2} - 500$

$= 500(500 + 1) - 500$

$= 500^2 + 500 - 500$

$= 250'000$

b. $2 + 4 + 8 + 16 + \cdots + 1024$

It's a geometric sequence that follows the pattern 2^i

* Found upper limit with $\log_2 1024 = 10$

$= \Sigma_{i=1}^{10} 2^i$

$= \Sigma_{i=0}^{10} 2^i - 2^0$ (subtract 2^0 to have it in the form of $\Sigma_{i=0}^{10} 2^i$)

$= (2^{10+1} - 1) - 1$ (use formula $\Sigma_{i=0}^{n} 2^i = 2^{n+1} - 1$)

$= 2^{11} - 2$

$= 2048 - 2 = 2046$

c. $\Sigma_{i=3}^{n+1} 1$

$= (n + 1) - 3 + 1$ (using $upper - lower + 1$)

$= n - 1$

d. $\Sigma_{i=3}^{n+1} i$

$= \Sigma_{i=1}^{n+1} i - \Sigma_{i=1}^{2} i$ (split sum to turn $\Sigma_{i=3}^{n+1} i$ into $\Sigma_{i=1}^{n+1} i$)

$= \frac{(n+1)(n+2)}{2} - \frac{2(2+1)}{2}$

$= \frac{(n+1)(n+2)-6}{2}$

$= \frac{n^2+3n+2-6}{2}$

$= \frac{n^2+3n-4}{2} = \frac{1}{2}(n^2 + 3n - 4)$

e. $\Sigma_{i=0}^{n-1} i(i + 1)$

$= \Sigma_{i=0}^{n-1}(i^2 + i)$

$= \Sigma_{i=0}^{n-1}(i^2) + \Sigma_{i=0}^{n-1}(i)$ (split sum)

$= (\frac{(n-1)(n)(2n-1)}{6}) + (\frac{(n-1)(n)}{2})$ (used summation properties)

$= (\frac{2n^3-3n^2+n}{6}) + (\frac{n^2-n}{2})$

$= \frac{1}{3}n^3 - \frac{1}{2}n^2 + \frac{1}{6}n + \frac{1}{2}n^2 - \frac{1}{2}n$

$= \frac{1}{3}n^3 - \frac{1}{3}n$

f. $\Sigma_{j=1}^{n} 3^{j+1}$

$= \Sigma_{j=1}^{n} 3^j 3^1$

$= 3 \cdot \Sigma_{j=1}^{n} 3^j$

$= 3(\frac{3^{n+1}-1}{3-1} - 1)$ (replace $\Sigma_{j=1}^{n} 3^j$ with $\frac{3^{n+1}-1}{3-1}$; additional –1 to account for 3^0)

$= 3(\frac{3^{n+1}-1}{2} - \frac{2}{2})$

$= 3(\frac{3^{n+1}-3}{2})$

$= \frac{3}{2}(3^{n+1} - 3)$

$= \frac{9}{2}(3^n - 1)$ (pull out additional –3)

g. $\Sigma_{i=1}^{n}\Sigma_{j=1}^{n} ij$

$= \Sigma_{i=1}^{n}(i \cdot \Sigma_{j=1}^{n} j)$

$= \Sigma_{i=1}^{n}(i \cdot \frac{n(n+1)}{2})$

$= \frac{n(n+1)}{2} \cdot \Sigma_{i=1}^{n}(i)$

$= \frac{n(n+1)}{2} \cdot \frac{n(n+1)}{2}$

$= \frac{1}{4}(n^2 + n)(n^2 + n)$

$= \frac{1}{4}(n^4 + 2n^3 + n^2)$

h. $\Sigma_{i=1}^{n} 1/(i(i+1))$

$= \Sigma_{i=1}^{n}(\frac{1}{i}) - \Sigma_{i=1}^{n}(\frac{1}{i+1})$ (fraction split)

$\approx (\ln n + \gamma) - \Sigma_{i=1}^{n}(\frac{1}{i+1})$ (approx. $\Sigma_{i=1}^{n}(\frac{1}{i}) \approx \ln x + \gamma$)

(*where $\gamma \approx 0.577215665$ (Euler's constant))

Let $k = i + 1$ therefore…

when $i = 1$, $k = 2$ (new lower limit)

when $i = n$, $k = n + 1$ (new upper limit)

$= (\ln n + \gamma) - \Sigma_{k=2}^{n+1}(\frac{1}{k})$ (substitute $i + 1$ with k)

$= (\ln n + \gamma) - (\Sigma_{k=1}^{n+1}(\frac{1}{k}) - (\frac{1}{1}))$ (subtract out $\frac{1}{1}$ so lower limit can be 1 not 2)

$\approx (\ln n + \gamma) - (\ln(n+1) + \gamma - 1)$

$= \ln n + \gamma - \ln(n+1) - \gamma + 1$

$= \ln n - \ln(n+1) + 1$

$= \ln(\frac{n}{n+1}) + 1$

---

4. Find the order of growth of the following sums. Use the (g(n)) notation with the simplest function g(n) possible.

a. $\Sigma_{i=0}^{n-1}(i^2 + 1)^2$

$= \Sigma_{i=0}^{n-1}(i^2 + 1)(i^2 + 1)$

$= \Sigma_{i=0}^{n-1}(i^4 + 2i^2 + 1)$

$= \Sigma_{i=0}^{n-1}(i^4) + \Sigma_{i=0}^{n-1}(2i^2) + n$

$= \Sigma_{i=0}^{n-1}(i^4) + 2 \cdot \Sigma_{i=0}^{n-1}(i^2) + n$

$\approx \frac{1}{4+1}(n-1)^{4+1} + 2(\frac{(n-1)(n)(2n-1)}{6}) + n$ (approx. $\Sigma_{i=0}^{n-1}(i^4) \approx \frac{1}{4+1}(n-1)^{4+1}$)

$= \frac{1}{5}(n-1)^5 + (\frac{2n^3 - 3n^2 + n}{3}) + n$

$= \frac{1}{5}(n-1)^5 + (\frac{2n^3 - 3n^2 + n}{3}) + n$

$= \frac{1}{5}(n-1)^5 + (\frac{2}{3}n^3 - n^2 + \frac{1}{3}n) + n$

$= \frac{1}{5}(n-1)^5 + \frac{2}{3}n^3 - n^2 + \frac{4}{3}n$

$(\frac{1}{5}(n-1)^5 + \frac{2}{3}n^3 - n^2 + \frac{4}{3}n) \approx (\frac{1}{5}n^5 + \frac{2}{3}n^3 - n^2 + \frac{4}{3}n) \approx n^5 + n^3 - n^2 + n$

$(n^5 + n^3 - n^2 + n) \in \Theta(max\{n^5, n^3, n^2, n\}) = \Theta(n^5)$

Therefore, the function $\frac{1}{5}(n-1)^5 + \frac{2}{3}n^3 - n^2 + \frac{4}{3}n \in \Theta(n^5)$

b. $\Sigma_{i=2}^{n-1} \lg i^2$

$= \Sigma_{i=2}^{n-1} 2 \lg i$

$= 2 \cdot \Sigma_{i=2}^{n-1} \lg i$

$= 2 \cdot (\Sigma_{i=1}^{n-1} \lg i - \Sigma_{i=1}^{1} \lg i)$

$= 2 \cdot (\Sigma_{i=1}^{n-1} \lg i - 0)$

$= 2(\Sigma_{i=1}^{n} \lg i - \lg n)$ (subtract $\lg n$, so $\Sigma_{i=1}^{n-1}$ can be substituted for $\Sigma_{i=1}^{n}$)

$\approx 2(n \lg n - \lg n)$ (substitute $\Sigma_{i=1}^{n} \lg i$ for the approximation $n \lg n$)

$\approx 2(\lg n^n - \lg n)$

$\approx 2(\lg \frac{n^n}{n})$

$\approx 2 \lg n^{n-1}$

$\approx 2(n - 1) \lg n$

$\approx (n - 1) \lg(n^2)$

As $n \to \infty$**, the function** $(n - 1) \lg(n^2) = 2(n - 1) \lg(n) \approx n \log(n^2)$

Therefore the function $(n - 1) \lg(n^2) \in \Theta(n \log n)$

c. $\Sigma_{i=1}^{n}(i + 1)2^{i-1}$

$= \Sigma_{i=1}^{n}(i + 1)2^{i}2^{-1}$

$= 2^{-1} \cdot \Sigma_{i=1}^{n}(i2^i + 2^i)$

$= 2^{-1}\Sigma_{i=1}^{n}(i2^i) + 2^{-1}\Sigma_{i=1}^{n}(2^i)$

$= \frac{1}{2}((n - 1)2^{n+1} + 2) + \frac{1}{2}(2^{n+1} - 1 - 1)$ (Minus a 2nd 1 since ($\Sigma_{i=1}^{n} 2^i$ not $\Sigma_{i=0}^{n} 2^i$))

$= (n - 1)2^n + 1 + 2^n - 1$

$= (n - 1)2^n + 2^n$

$= n2^n - 2^n + 2^n$

$= n2^n$

Therefore, the function $n2^n \in \Theta(n2^n)$

d. $\Sigma_{i=0}^{n-1}\Sigma_{j=0}^{i-1}(i + j)$

$= \Sigma_{i=0}^{n-1}(\Sigma_{j=0}^{i-1}(i) + \Sigma_{j=0}^{i-1}(j))$ (split sum)

$= \Sigma_{i=0}^{n-1}(i(i) + \frac{(i-1)(i)}{2})$

$= \Sigma_{i=0}^{n-1}(i^2 + \frac{1}{2}i^2 - \frac{1}{2}i)$

$= \Sigma_{i=0}^{n-1}(\frac{3}{2}i^2 - \frac{1}{2}i)$

$= \frac{3}{2} \cdot \Sigma_{i=0}^{n-1}(i^2) - \frac{1}{2} \cdot \Sigma_{i=0}^{n-1}(i)$

$= \frac{3}{2}(\frac{(n-1)(n)(2n-1)}{6}) - \frac{1}{2}(\frac{(n-1)(n)}{2})$

$= (\frac{(n-1)(n)(2n-1)}{4}) - (\frac{(n-1)(n)}{4})$

$= \frac{1}{4}[(n-1)(n)(2n-1) - (n-1)(n)]$

$= \frac{1}{4}(n-1)[(n)(2n-1) - (n)]$

$= \frac{1}{4}(n-1)[2n^2 - 2n]$

$= \frac{1}{4}2n(n-1)(n-1)$

$= \frac{1}{2}n(n-1)^2$

As $n \to \infty$, the function $\frac{1}{2}n(n-1)^2 \approx \frac{1}{2}n(n)^2$

Therefore, $\frac{1}{2}n(n-1)^2 \in \Theta(n^3)$

---

5. The sample variance of n measurements $x_1, ..., x_n$ can be computed as either

$V = \frac{1}{n-1}\Sigma_{i=1}^{n}(x_i - \bar{x})^2$ where $\bar{x} = \frac{1}{n}\Sigma_{i=1}^{n}x_i$

Find and compare the number of divisions, multiplications, and additions/subtractions (additions and subtractions are usually bunched together) that are required for computing the variance according to each of these formulas.

	Adding/Subtracting	Multiplication	Division
Variance	$AS_V(n) = A_V(n) + S_V(n)$	$M_V(n)$	$D_V(n)$
Mean	$AS_{\bar{x}}(n) = A_{\bar{x}}(n) + S_{\bar{x}}(n)$	$M_{\bar{x}}(n)$	$D_{\bar{x}}(n)$

First solve for when variance and mean are separated, so $V = \frac{\Sigma_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$ where $\bar{x} = \frac{\Sigma_{i=1}^{n}x_i}{n}$. I get the results:

$AS_V(n) = (n-1) + n + 1 = 2n$

- $\Sigma_{i=1}^{n}(...)$ adds $(n-1)$ times, $(x_i - \bar{x})$ is inside $\Sigma_{i=1}^{n}$ so is calculated n times, the denominator $\frac{...}{n-1}$ is calculated once.

$AS_{\bar{x}}(n) = n - 1$

- $\Sigma_{i=1}^{n}x_i$ adds $(n-1)$ times

$M_V(n) = n$

- There is a square inside $\Sigma_{i=1}^{n}$, so it is calculated n times.

$M_{\bar{x}}(n) = 0$

- There is no multiplication in the mean formula.

$D_V(n) = 1$

- There is 1 division in the variance formula.

$D_{\bar{x}}(n) = 1$

- There is 1 division in the mean formula.

If the formulas are independent then we get:
$AS(n) = AS_V(n) + AS_{\bar{x}}(n) = (2n + n - 1) = (3n - 1)$

$M(n) = M_V(n) + M_{\bar{x}}(n) = (n + 0) = n$

$D(n) = D_V(n) + D_{\bar{x}}(n) = (1 + 1) = 2$

Next solve when they are together (unoptimized):

$$V = \frac{\Sigma_{i=1}^{n}(x_i - \frac{\Sigma_{i=1}^{n} x_i}{n})^2}{n - 1}$$

* All mean operations are multiplied by n since they are executing every iteration.

Solve Addition/Subtraction:
$AS(n) = AS_V(n) + n \cdot AS_{\bar{x}}(n)$

$AS(n) = 2n + n(n - 1)$

$AS(n) = 2n + n^2 - n$

$AS(n) = n^2 + n$

Solve Multiplication:
$M(n) = M_V(n) + n \cdot M_{\bar{x}}(n)$

$M(n) = n + n(0)$

$M(n) = n$

Solve Division:
$D(n) = D_V(n) + n \cdot D_{\bar{x}}(n)$

$D(n) = 1 + n(1)$
$D(n) = 1 + n$

If the two formulas are separated we get $AS(n) = 3n - 1$**,** $M(n) = n$**, and** $D(n) = 2$**. However, if we nest the formula for mean inside the formula for variance we get** $AS(n) = n^2 + n$**,** $M(n) = n$**, and** $D(n) = 1 + n$

---

6. Solve the following recurrences and find the efficiency class for each of them.
   (1) $A(n) = 3A(n - 1)$ for $n > 1$, $A(1) = 4$

   $A(n - 1) = 3[3A((n - 1) - 1)] = 3^2A(n - 2)$

   $A(n - 2) = 3^2[3A((n - 2) - 1)] = 3^3A(n - 3)$

   $A(n - 3) = 3^3[3A((n - 3) - 1)] = 3^4A(n - 4)$

   When looking at the iterations, there is a clear pattern that emerges:

   $A(n) = 3^iA(n - i)$

We will reach our initial condition when $n - i = 1$ or $i = n - 1$
When $i = n - 1,\ \ A(n) = 4$
$A(n) = 3^{n-1}A(n - (n - 1))$ (substitute i with $n - 1$)
$A(n) = \frac{1}{3}3^{n}A(n - n + 1)$
$A(n) = \frac{1}{3}3^{n}A(1)$
$A(n) = \frac{1}{3}3^{n}[4]$ (substitute $A(1)$ with 4)
$A(n) = \frac{4}{3}3^{n}$

The function $\frac{4}{3}3^{n} \in \Theta(3^{n})$
The algorithm is in the efficiency class $\Theta(3^{n})$

(2) $A(n) = A(n - 1) + 5$ for $n > 1,\ A(1) = 0$
$A(n - 1) = [A((n - 1) - 1) + 5] + 5 = A(n - 2) + 10$
$A(n - 2) = [A((n - 2) - 1) + 5] + 10 = A(n - 3) + 15$
$A(n - 3) = [A((n - 3) - 1) + 5] + 15 = A(n - 4) + 20$

When looking at the iterations, there is a clear pattern that emerges for $n > 1$:
$$A(n) = A(n - i) + 5i$$
We will reach our initial condition when $n - i = 1$ or $i = n - 1$:
$A(n) = A(n - (n - 1)) + 5(n - 1)$ (substitute i with $n - 1$)
$A(n) = A(n - n + 1) + 5n - 5$
$A(n) = A(1) + 5n - 5$
$A(n) = [0] + 5n - 5$ (substitute $A(1)$ with 0)
$A(n) = 5n - 5$

As $n \to \infty$, $5n - 5 \approx 5n \in \Theta(n)$
The algorithm is in the efficiency class $\Theta(n)$

(3) $A(n) = A(n - 1) + n$ for $n > 0,\ A(0) = 0$
$A(n) = [A((n - 1) - 1) + (n - 1)] + n = A(n - 2) + 2n - 1$
$A(n) = [A((n - 2) - 1) + (n - 2)] + 2n - 1 = A(n - 3) + 3n - 3$
$A(n) = [A((n - 3) - 1) + (n - 3)] + 3n - 3 = A(n - 4) + 4n - 6$
$A(n) = [A((n - 4) - 1) + (n - 4)] + 4n - 6 = A(n - 5) + 5n - 10$

The integer seems to follow the trend of $1,\ 3,\ 6,\ 10$ … Like that of $\Sigma_{k=0}^{n}k = \frac{n(n+1)}{2}$
When looking at the iterations, there is a clear pattern that emerges for $n > 0$:
$$A(n) = A(n - i) + in - \Sigma_{k=0}^{i-1}k$$
We will reach our initial condition when $n - i = 0$ or $n = 0$:
$A(n) = A(n - i) + in - \frac{(i-1)(i)}{2}$ (substitute $\Sigma_{k=0}^{i-1}k$ with $\frac{(i-1)(i)}{2}$)

$A(n) = A(n - i) + in - \frac{1}{2}(i^2 - i)$

$A(n) = A(n - (n)) + (n)n - \frac{1}{2}((n)^2 - (n))$ (substitute i with n)

$A(n) = A(0) + n^2 - \frac{1}{2}n^2 + \frac{1}{2}n$

$A(n) = [0] + \frac{1}{2}n^2 + \frac{1}{2}n$

$A(n) = \frac{1}{2}n^2 + \frac{1}{2}n$

The function $\frac{1}{2}n^2 + \frac{1}{2}n \in \Theta(n^2) + \Theta(n) = \Theta(max\{n^2, n\}) = \Theta(n^2)$

Therefore the algorithm is in the efficiency class $\Theta(n^2)$.

(4) $A(n) = A(n/5) + 1$ for $n > 1$, $A(1) = 1$ (solve for $n = 5^k$)

$A(5^k) = A(5^{k-1}) + 1$

$= [A(5^{(k-1)-1}) + 1] + 1 = A(5^{k-2}) + 2$

$= [A(5^{(k-2)-1}) + 1] + 2 = A(5^{k-3}) + 3$

When looking at the iterations, there is a clear pattern that emerges:

$A(5^{k-i}) + i$

We will reach the initial condition when $5^{k-i} = 1$ or when $i = k$:

General Formula: $A(5^{k-i}) + i$:

$A(5^k) = A(5^{k-k}) + k$ (substitute i for k)

$A(5^k) = A(5^0) + k$

$A(5^k) = A(1) + k$

$A(5^k) = [1] + k$ (substitute $A(1)$ with 1)

$A(n) = 1 + \log_5 n$ (substitute k with $\log_5 n$)

As $n \to \infty$, the function $1 + \log_5 n \approx \log_5 n$.

The algorithm is in the efficiency class $\Theta(\log n)$

(5) $A(n) = 2A(n/2) + n - 1$ for $n > 1$, $A(1) = 0$ (solve for $n = 2^k$)

Recursive step:

$= 2A(2^{k-1}) + 2^k - 1$

$= 2[2A(2^{k-2}) + 2^{k-1} - 1] + 2^k - 1 = 2^2A(2^{k-2}) + 2(2^k) - 1 - 2$

$= 2^2[2A(2^{k-3}) + 2^{k-2} - 1] + 2(2^k) - 3 = 2^3A(2^{k-3}) + 3(2^k) - 1 - 2 - 4$

$= 2^3[2A(2^{k-4}) + 2^{k-3} - 1] + 3(2^k) - 7 = 2^4A(2^{k-4}) + 4(2^k) - 1 - 2 - 4 - 8$

When looking at the iterations a clear pattern emerges:

$A(2^k) = 2^iA(2^{k-i}) + i(2^k) - \Sigma_{j=0}^{i-1}(2^j)$

The base condition will occur when $2^{k-i} = 1$ or when $i = k$:

$A(2^k) = 2^kA(2^{k-k}) + k(2^k) - \Sigma_{j=0}^{k-1}(2^j)$ (substitute i with k)

$A(2^k) = 2^k A(2^0) + k2^k - (2^{(k-1)+1} - 1)$

$A(2^k) = 2^k A(1) + k2^k - 2^k + 1$

$A(2^k) = 2^k[0] + k2^k - 2^k + 1$ (substitute $A(1)$ with 0)

$A(2^k) = 0 + 2^k(k - 1) + 1$

$A(2^k) = 2^k(k - 1) + 1$

$A(n) = n(\log_2 n - 1) + 1$

$A(n) = n\log_2 n - n + 1$ (substitute k with $\log_2 n$)

$n\log_2 n - n + 1 \approx n\log_2 n - n \in \Theta(n \log n) + \Theta(n)$

$= \Theta(max\{n \log n,\ n\}) = \Theta(n \log n)$

The algorithm is in the efficiency class $\Theta(n \log n)$.

---

7. Consider the following algorithm and answer the questions.

ALGORITHM X($A[0..n - 1]$)
 // Input: A contains n real numbers
 for $i \leftarrow 0$ to $n - 2$ do
  for $j \leftarrow i + 1$ to $n - 1$ do
   if $A[j] > A[i]$
    swap $A[i]$ and $A[j]$

(1) What does this algorithm compute?

This algorithm sorts the given array from highest to lowest. I can tell this from a number of factors: the input type being an array, the nested for loops which iterate through the list elements, and the condition of which checks if the next element is greater than the previous, and if so, swaps the 2 elements.

(2) What is the input size?

The input size is based on the size of the array, or the value of (n).

(3) What is the basic operation?

The basic operation is the comparison between 2 elements in the array. It occurs in every iteration of the loop. This is further evident with the fact that this algorithm is for sorting.

(4) How many times is the basic operation executed? (Set up a sum, and simplify the sum, to find the function expressing the number of repetitions of the basic operation.)

The outer loop can be represented with $\Sigma_{i=0}^{n-2}$, and the inner loop can be represented with $\Sigma_{j=i+1}^{n-1}$. Since the comparison happens once within the inner loop, we end up with $\Sigma_{i=0}^{n-2}\Sigma_{j=i+1}^{n-1}(1)$.

$\Sigma_{i=0}^{n-2}\Sigma_{j=i+1}^{n-1}(1)$

$= \Sigma_{i=0}^{n-2}[(n - 1) - (i + 1) + 1)]$ $(upper - lower + 1)$
$= \Sigma_{i=0}^{n-2}(n - 1 - i - 1 + 1)$
$= \Sigma_{i=0}^{n-2}(n - i - 1)$
$= \Sigma_{i=0}^{n-2}(n) - \Sigma_{i=0}^{n-2}(1) - \Sigma_{i=0}^{n-2}(i)$
$= n \cdot \Sigma_{i=0}^{n-2}(1) - \Sigma_{i=0}^{n-2}(i) - \Sigma_{i=0}^{n-2}(1)$
$= n(n - 1) - \frac{(n-2)(n-1)}{2} - (n - 1)$
$= n^2 - n - \frac{1}{2}n^2 + \frac{3}{2}n - 1 - n + 1$
$= \frac{1}{2}n^2 - \frac{1}{2}n$

(5) What is the efficiency class of this algorithm?

$\frac{1}{2}n^2 - \frac{1}{2}n \in \Theta(n^2) - \Theta(n) = \Theta(n^2)$

The efficiency class of the algorithm is $\Theta(n^2)$. Furthermore, it does not contain a best and worst case since the algorithm will execute no matter the given input (even if they are all sorted).

---

8. Consider the following algorithm and answer the questions.

ALGORITHM Y(n)
// Input: n is a positive integer
if $n = 1$ return 1
else return $Y(n - 1) + n * n$

(1) What does this algorithm compute?

The function computes the sum of squares from 1 to n.

(2) What is the input size?

The input size is based on n, where $n \geq 1$.

(3) What is the basic operation?

The basic operation of the algorithm is the addition of each square.

(4) Set up a recurrence and an initial condition and find the number of times the basic operation is executed.

Recurrence $A(n) = A(n - 1) + 1$, with initial condition: $A(1) = 0$

$A(n) = A(n - 1) + 1$
$= [A((n - 1) - 1) + 1] + 1 = A(n - 2) + 2$
$= [A((n - 2) - 1) + 1] + 2 = A(n - 3) + 3$
$= [A((n - 3) - 1) + 1] + 3 = A(n - 4) + 4$

Basic pattern of $A(n) = A(n - i) + i$, will reach initial condition when $n - i = 1$ or when $i = n - 1$:

$A(n) = A(n - (n - 1)) + (n - 1)$

$A(n) = A(n - n + 1) + n - 1$
$A(n) = A(1) + n - 1$
$A(n) = [0] + n - 1$
$A(n) = n - 1$

(5) What is the efficiency class of this algorithm?

As $n \to \infty$ the function $n - 1 \approx n$

Therefore $n - 1 \in \Theta(n)$

The efficiency class for the Algorithm $Y(n)$ is $\Theta(n)$. The algorithm does not have a best/worst case since the program will execute all the way through for any given value of n (i.e. there is no chance for it to end early).

9. Consider the following algorithm and answer the questions.

```
ALGORITHM T(A[0..n − 1])
    // Input: A contains n ≥ 1 real numbers
    for i ← 1 to n − 1 do
        v ← A[i]
        j ← i − 1
        while j ≥ 0 and A[j] > v do
            A[j + 1] ← A[j]
            j ← j − 1
        A[j + 1] ← v
```

* First change algorithm to have a for loop instead of a while:

```
ALGORITHM T(A[0..n − 1])
    / Input: A contains n ≥ 1 real numbers
    for i ← 1 to n − 1 do
        v ← A[i]
        j ← i − 1
        for j ← i − 1 downto 0 do
            if A[j] > v
                break
            A[j + 1] ← A[j]
        A[j + 1] ← v
```

(1) What does this algorithm compute?

The algorithm is a sorting algorithm that would sort elements in ascending order. I can tell this from 3 main things: the nested loop iterating through each element, the rearrangement of elements, and the end condition of $A[j] > v$.

(2) What is the input size?

The input size is dependant on the size of the array (n).

(3) What is the basic operation?

Because this is a sorting algorithm, the basic operation is comparing, specifically the comparison ($A[j] > v$).

(4) How many times is the basic operation executed? (Set up a sum, and simplify the sum, to find the function expressing the number of repetitions of the basic operation.)

The basic operation is executed once* inside the inner loop.

$\Sigma_{i=1}^{n-1}\Sigma_{i=0}^{i-1}(1)$

$= \Sigma_{i=1}^{n-1}(i)$ (use property $\Sigma_{i=0}^{n-1}(1) = n$)

$= \frac{(n-1)(n)}{2}$ (use property $\Sigma_{i=0}^{n} i = \frac{n(n+1)}{2}$)

$= \frac{n^2-n}{2}$

$= \frac{1}{2}(n^2 - n)$

(5) What is the efficiency class of this algorithm?

As $n \to \infty$, the function $\frac{1}{2}(n^2 - n) = \frac{1}{2}n(n - 1) \approx \frac{1}{2}n(n) \approx n^2$

Therefore $\frac{1}{2}(n^2 - n) \in \Theta(max\{n^2, n\})$ or $\frac{1}{2}(n^2 - n) \in \Theta(n^2)$

As there is a way for this algorithm to stop part way through execution, it will have a best, average and worst case. The worst case for this algorithm is in $O(n^2)$ and is in the efficiency class $\Theta(n^2)$.

---

10. Consider the following algorithm and answer the questions.

```
ALGORITHM W(A, l, r, K)
    // Input: A is an array of sorted integers,
    //        l and r are the leftmost and rightmost indexes of the
    //        array elements to be processed,
    //        K is an integer
    if l > r return –1
    else
        m ← ⌊(l + r)/2⌋
        if K = A[m] return m
        else if K < A[m] return W(A, l, m − 1, K)
            else return W(A, m + 1, r, K)
```

(1) What does the algorithm compute?

Since the algorithm requires A to be a sorted list of integers, as well as the fact that it either searches either the top or bottom half of the array (depending on the position of K) this algorithm is a binary search algorithm.

(2) How is the input size n expressed in terms of the parameters?

Since the size of the array A is not explicitly expressed, we have to look at the left (l) and right (r) pointers, as they determine the size of the array. The size n can be expressed as $n = (r - l) + 1$ (the +1 accounts for the loss in subtraction)

For example if $l = 2$ and $r = 5$, then $n = (5 - 2) + 1 = 4$.

However, since the middle index m (± 1) will become either l or r on each initial pass, the size of A will be cut in half with each pass ($n = \lfloor \frac{n}{2} \rfloor$)

(3) Assume that after comparison of K with $A[m]$, the algorithm can determine whether K is smaller than, equal to, or larger than $A[m]$. Set up a recurrence (with an initial condition) for comparison in the worst case of this algorithm. Solve the recurrence for $n = 2^k$, and determine the Θ efficiency class.

Since the basic $C(n) = C(\lfloor \frac{n}{2} \rfloor) + 1$
In the worst case, our initial condition will occur when $n = 1$, and 0 comparisons will occur in the worst case, the element won't be in the array)
Recurrence: $C(n) = C(\lfloor \frac{n}{2} \rfloor) + 1$ for $(n > 1)$
Initial Condition: $C(1) = 1$

Assuming $n = 2^k$ solve the recurrence step:

$$
\begin{aligned}
C(2^k) &= C(2^{k-1}) + 1 \\
&= [C(2^{(k-1)-1}) + 1] + 1 = C(2^{k-2}) + 2 \\
&= [C(2^{(k-2)-1}) + 1] + 2 = C(2^{k-3}) + 3 \\
&= [C(2^{(k-3)-1}) + 1] + 3 = C(2^{k-4}) + 4
\end{aligned}
$$

When looking at the iterations, there is a clear pattern that emerges:

$$C(2^k) = C(2^{k-i}) + i$$

We will reach our initial condition when $2^{k-i} = 1$ or $i = k$:

$C(2^k) = C(2^{k-k}) + k$ (substitute i for k)
$C(2^k) = C(2^0) + k$
$C(2^k) = C(1) + k$
$C(2^k) = [0] + k$ (substitute $C(1)$ for 0)
$C(2^k) = k$
$C(n) = \log_2 n$ (substitute k for $\log_2 n$)

$\log_2 n \in \Theta(\log n)$
The algorithm is in the efficiency class $\Theta(\log n)$
$C(2^k) = C(2^{k-1}) + 1$

(4) What is the Θ efficiency class when $n \neq 2^k$? Why?

The Θ efficiency class of the algorithm $T(\ldots)$ when $n \neq 2^k$ is still $\Theta(\log n)$. The reason for this can be explained by the smoothness rule; because the $\log_2 n$ is a non-decreasing function, and that $\log_2 n \in \Theta(\log n)$ is true for all values of $n = 2^k$, then $\log_2 n \in \Theta(\log n)$ will still be true for any value of n even when $n \neq 2^k$.

Example:

$f(5n)$, where $f(n) = \log_2 n$

$$\log_2(5n) = \log_2 5 + \log_2 n$$

$$\approx (2.322 + \log_2 n) \in \Theta(\log n)$$