Problem 1. Prove $\log_2(n!) = \Theta(n \log n)$.

Solution. Let us prove first $\log_2(n!) = O(n \log n)$:

$$
\log_2(n!) = \log_2(\Pi_{i=1}^{n} i) \\
\leq \log_2 n^n \\
= n \log_2 n \\
= O(n \log n).
$$

Now we prove $\log_2(n!) = \Omega(n \log n)$:

$$
\log_2(n!) = \log_2(\Pi_{i=1}^{n} i) \\
\geq \log_2(\Pi_{i=n/2}^{n} i) \\
\geq \log_2(n/2)^{n/2} \\
= (n/2) \log_2(n/2) \\
= \Omega(n \log n).
$$

This completes the proof.

Problem 2. Let $f(n)$ be a function of positive integer $n$. We know:

$$
\begin{align*}
  f(1) &= 1 \\
  f(n) &\leq 2 + f([n/10]).
\end{align*}
$$

Prove $f(n) = O(\log n)$. Recall that $[x]$ is the ceiling operator that returns the smallest integer at least $x$.

Solution 1 (Expansion). Consider first $n$ being a power of 10.

$$
\begin{align*}
  f(n) &\leq 2 + f(n/10) \\
  &\leq 2 + 2 + f(n/10^2) \\
  &\leq 2 + 2 + 2 + f(n/10^3) \\
  &\vdots \\
  &\leq 2 \cdot \log_{10} n + f(1) \\
  &= 2 \cdot \log_{10} n + 1 = O(\log n).
\end{align*}
$$

Now consider $n$ that is not a power of 10. Let $n'$ be the smallest power of 10 that is greater than $n$. We have:

$$
\begin{align*}
  f(n) &\leq f(n') \\
  &\leq 2 \log_{10} n' + 1 \\
  &\leq 2 \log_{10} (10n) + 1 \\
  &\leq O(\log n).
\end{align*}
$$
Solution 2 (Master Theorem). Let $\alpha, \beta$, and $\gamma$ be as defined in the Master Theorem (see the tutorial slides of Week 4). Thus, we have $\alpha = 1$, $\beta = 10$, and $\gamma = 0$. Since $\log_\beta \alpha = \log_{10} 1 = 0 = \gamma$, by the Master Theorem, we know that $f(n) = O(n^\gamma \log n) = O(\log n)$.

Solution 3 (Substitution). We aim to prove that $f(n) \leq 1 + \alpha \log_2 n$ for some constant $\alpha$ to be chosen later. Let $\beta \geq 1$ be another constant that will also be decided later.

Base case ($n \leq \beta$). For every $n \in [1, \beta]$, we need

$$f(n) \leq 1 + \alpha \log_2 n$$

(1)

The above definitely holds when $n = 1$. For $n \in [2, \beta]$, we will need

$$\alpha \geq \frac{f(n) - 1}{\log_2 n}.$$  

(2)

Inductive case. Assuming $f(n) \leq 1 + \alpha \log_2 n$ for all $n \leq t - 1$ where $t \geq \beta + 1$, we want to prove $f(t) \leq 1 + \alpha \log_2 t$.

We will consider only

$$\beta \geq 2$$

(3)

such that $t \geq 3$ and, hence, $[t/10] \leq (t/10) + 1 \leq t/2$. With this, we have:

$$f(t) \leq 2 + f([t/10])$$

$$\leq 3 + \alpha \log_2 [t/10]$$

$$\leq 3 + \alpha \log_2 (t/2)$$

$$= 3 + \alpha \log_2 t - \alpha.$$

To complete the inductive argument, we need the above to be at most $1 + \alpha \log_2 t$, namely:

$$\alpha \geq 2.$$  

(4)

To satisfy (2)-(4), we set $\beta = 2$ and $\alpha = \max\{2, (f(2) - 1)/\log_2 2\} = \max\{2, f(2) - 1\}$.

Problem 3. Let $f(n)$ be a function of positive integer $n$. We know:

$$f(1) = 1$$

$$f(n) \leq 2 + f([3n/10]).$$

Prove $f(n) = O(\log n)$. Recall that $[x]$ is the ceiling operator that returns the smallest integer at least $x$.

Solution 1 (Expansion).

$$f(n) \leq 2 + f(n_1) \quad (\text{define } n_1 = [(3/10)n])$$

$$f(n) \leq 2 + 2 + f(n_2) \quad (\text{define } n_2 = [(3/10)n_1])$$

$$f(n) \leq 2 + 2 + 2 + f(n_3) \quad (\text{define } n_3 = [(3/10)n_2])$$

$$\vdots$$

$$f(n) \leq 2 + 2 + \ldots + 2 + f(n_h) \quad (\text{define } n_h = [(3/10)n_{h-1}])$$

$$= 2h + f(n_h).$$  

(5)
So it remains to analyze the value of \( h \) that makes \( n_h \) small enough. Note that we do not need to solve the precise value of \( h \); it suffices to prove an upper bound for \( h \). For this purpose, we reason as follows. First, notice that

\[
\left\lceil \frac{3n}{10} \right\rceil \leq \frac{4n}{10}
\]

when \( n \geq 10 \) (prove this yourself).

Let us set \( h \) to be the smallest integer such that \( n_h < 10 \) (this implies that \( n_{h-1} \geq 10 \) and \( n_h \geq (4/10)n_{h-1} \geq 4 \)). We have:

\[
\begin{align*}
n_1 &\leq (4/10)n \\
n_2 &= \left\lceil (3/10)n_1 \right\rceil \leq (4/10)n_1 \leq (4/10)^2 n \\
n_3 &\leq (4/10)^3 n \\
&\vdots \\
n_h &\leq (4/10)^h n
\end{align*}
\]

Therefore, the value of \( h \) cannot exceed \( \log_{10} 4n \) (otherwise, \((4/10)^4 \cdot n < 1\), making \( n_h < 1 \), which contradicts the fact that \( n_h \geq 4 \)). Plugging this into (5) gives:

\[
f(n) \leq 2 \log_{10} n + f(10) = O(\log n). \quad \text{(think: why?)}
\]

**Solution 2 (Master Theorem).** Let \( \alpha, \beta, \) and \( \gamma \) be as defined in the Master Theorem. Thus, we have \( \alpha = 1, \beta = 10/3, \) and \( \gamma = 0 \). Since \( \log_{\beta} \alpha = \log_{10/3} 1 = 0 = \gamma \), by the Master Theorem, we know that \( f(n) = O(n^\gamma \log n) = O(\log n). \)

**Solution 3 (Substitution).** We aim to prove that \( f(n) \leq 1 + \alpha \log_2 n \) for some constant \( \alpha \) to be chosen later. Let \( \beta \geq 1 \) be another constant that will also be decided later.

**Base case** \((n \leq \beta)\). For \( n = 1 \), \( f(n) \leq 1 + \alpha \log_2 n \) always holds. For every \( n \in [2, \beta] \), we need

\[
\begin{align*}
f(n) &\leq 1 + \alpha \log_2 n \\
\iff \alpha &\geq \frac{f(n) - 1}{\log_2 n}.
\end{align*}
\]

**Inductive case.** Assuming \( f(n) \leq 1 + \alpha \log_2 n \) for all \( n \leq t - 1 \) where \( t \geq \beta + 1 \), we want to prove \( f(t) \leq 1 + \alpha \log_2 t \).

We will consider only

\[
\beta \geq 4
\]

such that \( t \geq 5 \) and, hence, \( \left\lceil 3t/10 \right\rceil \leq (3t/10) + 1 \leq t/2 \). With this, we have:

\[
\begin{align*}
f(t) &\leq 2 + f(\left\lceil 3t/10 \right\rceil) \\
&\leq 3 + \alpha \log_2 \left\lceil 3t/10 \right\rceil \\
&\leq 3 + \alpha \log_2 (t/2) \\
&= 3 + \alpha \log_2 t - \alpha.
\end{align*}
\]

To complete the inductive argument, we need the above to be at most \( 1 + \alpha \log_2 t \), namely:

\[
\alpha \geq 2.
\]
To satisfy (7)-(9), we set $\beta = 4$ and $\alpha = \max\{2, f(2) - 1, \frac{f(3)-1}{\log_2 3}, \frac{f(4)-1}{2}\}$.

**Problem 4.** Let $f(n)$ be a function of positive integer $n$. We know:

$$
\begin{align*}
  f(1) &= 1 \\
  f(n) &\leq 2n + 4f(\lfloor n/4 \rfloor).
\end{align*}
$$

Prove $f(n) = O(n \log n)$.

**Solution 1 (Expansion).** Consider first $n$ being a power of 4.

\[
\begin{align*}
  f(n) &\leq 2n + 4f(n/4) \\
  &\leq 2n + 4(2n/4 + 4f(n/4^2)) \\
  &\leq 2n + 2n + 4^2 f(n/4^2) \\
  &= 2 \cdot 2n + 4^2 f(n/4^2) \\
  &\leq 2 \cdot 2n + 4^2 \cdot (2(n/4^2) + 4f(n/4^3)) \\
  &= 3 \cdot 2n + 4^3 f(n/4^3) \\
  &\cdots \\
  &= (\log_4 n) \cdot 2n + n \cdot f(1) \\
  &= (\log_4 n) \cdot 2n + n = O(n \log n).
\end{align*}
\]

Now consider that $n$ is not a power of 4. Let $n'$ be the smallest power of 4 that is greater than $n$. This implies that $n' < 4n$. We have:

\[
\begin{align*}
  f(n) &\leq f(n') \\
  &\leq (\log_4 n') \cdot 2n' + n' \\
  &< (\log_4 (4n)) \cdot 8n + 4n = O(n \log n).
\end{align*}
\]

**Solution 2 (Master Theorem).** Let $\alpha, \beta,$ and $\gamma$ be as defined in the Master Theorem. Thus, we have $\alpha = 4, \beta = 4,$ and $\gamma = 1$. Since $\log_\beta \alpha = \log_4 4 = 1 = \gamma$, by the Master Theorem, we know that $f(n) = O(n^\gamma \log n) = O(n \log n)$.

**Solution 3 (Substitution).** We aim to prove that $f(n) \leq 1 + \alpha n \log_2 n$ for some constant $\alpha$ to be chosen later. Let $\beta \geq 1$ be another constant that will also be decided later.

**Base case** ($n \leq \beta$). For $n = 1$, it always holds that $f(1) \leq 1 + \alpha \log_2 n$. For every $n \in [2, \beta]$, we need

\[
\begin{align*}
  f(n) &\leq 1 + \alpha n \log_2 n \\
  \Leftrightarrow \alpha &\geq \frac{f(n) - 1}{n \log_2 n}.
\end{align*}
\]

**Inductive case.** Assuming $f(n) \leq 1 + \alpha n \log_2 n$ for all $n \leq t - 1$ where $t \geq \beta + 1$, we want to prove $f(t) \leq 1 + \alpha t \log_2 t$.

We will consider only

\[
\beta \geq 4
\]
such that \( t \geq 5 \) and, hence, \( t/4 + 1 \leq t/2 \). With this, we have:

\[
\begin{align*}
    f(t) & \leq 2t + 4(1 + \alpha[t/4] \log_2[t/4]) \\
         & \leq 4 + 2t + 4\alpha(t/4 + 1) \log_2(t/4 + 1) \\
         & \leq 4 + 2t + 4\alpha(t/4 + 1) \log_2(t/2) \\
         & = 4 + 2t + (\alpha t + 4\alpha)(\log_2 t - 1) \\
         & \leq 4 + 2t + (\alpha + 4\alpha) \log_2 t - \alpha t - 4\alpha \\
         & \leq 4 + 2t + \alpha t \log_2 t + 4\alpha \log_2 t - \alpha t - 4\alpha
\end{align*}
\]

To complete the inductive argument, we need the above to be at most \( 1 + \alpha t \log_2 t \), namely:

\[
3 + 2t + 4\alpha \log_2 t \leq \alpha t + 4\alpha
\]  \hspace{1cm} (12)

We will make sure

\[
\beta \geq 2^8
\]  \hspace{1cm} (13)

Under the above condition, for any \( t \geq \beta \), it holds that \( \log_2 t \leq t/8 \). To ensure (12), we require:

\[
\begin{align*}
    3 + 2t + 4\alpha t/8 & \leq \alpha t + 4\alpha \\
    \iff 3 + 2t + \alpha t/2 & \leq \alpha t + 4\alpha \\
    \iff 3 + 2t & \leq \alpha t/2 + 4\alpha \\
    & \iff 3 + 2t \leq \alpha t/2 + 4\alpha \quad \text{(as \( t \geq \beta \geq 2^8 \))}
\end{align*}
\]  \hspace{1cm} (14)

To satisfy (10), (11), (13), and (14), we set \( \beta = 2^8 \) and \( \alpha = \max\{5, \frac{f(2)-1}{2}, \frac{f(3)-1}{3 \log_2 3}, \ldots, \frac{f(2^8)-1}{2^8 \cdot 8}\} \).

**Problem 5 (Bubble Sort).** Let us re-visit the sorting problem. Recall that, in this problem, we are given an array \( A \) of \( n \) integers, and need to re-arrange them in ascending order. Consider the following *bubble sort* algorithm:

1. If \( n = 1 \), nothing to sort; return.
2. Otherwise, do the following in ascending order of \( i \in [1, n - 1] \): if \( A[i] > A[i + 1] \), swap the integers in \( A[i] \) and \( A[i + 1] \).

Prove that the algorithm terminates in \( O(n^2) \) time.

As an example, support that \( A \) contains the sequence of integers \((10, 15, 8, 29, 13)\). After Step 2 has been executed once, array \( A \) becomes \((10, 8, 15, 13, 29)\).

**Solution.** Let \( f(n) \) be the worst-case running time of bubble sort when \( A \) has \( n \) elements. From Step 1, we know:

\[
f(1) = O(1).
\]

From Steps 2-3, we know:

\[
f(n) \leq f(n - 1) + O(n).
\]

Solving the recurrence (by the expansion method) gives \( f(n) = O(n^2) \).

**Problem 6* (Modified Merge Sort).** Let us consider a variant of the merge sort algorithm for sorting an array \( A \) of \( n \) elements (we will use the notation \( A[i..j] \) to represent the part of the array from \( A[i] \) to \( A[j] \):

5
• If \( n = 1 \) then return immediately.
• Otherwise set \( k = \lceil n/3 \rceil \).
• Recursively sort \( A[1..k] \) and \( A[k+1..n] \), respectively.
• Merge \( A[1..k] \) and \( A[k+1..n] \) into one sorted array.

Prove that this algorithm runs in \( O(n \log n) \) time.

Solution. Let \( f(n) \) be the worst case time of the algorithm on an array of size \( n \). We have the following recurrence:

\[
\begin{align*}
f(1) & \leq c' \\
f(n) & \leq f(\lceil n/3 \rceil) + f(\lceil 2n/3 \rceil) + c \cdot n
\end{align*}
\]

where \( c > 0 \) and \( c' > 0 \) are constants.

We will prove that \( f(n) \leq c' + \alpha \cdot n \log_2 n \) for some constant \( \alpha \) to be decided later. Let \( \beta \geq 1 \) be another constant that will also be decided later.

**Base case** \((n \leq \beta)\). For \( n = 1 \), \( f(n) \leq c' + \alpha \cdot n \log_2 n \) always holds. For \( n \in [2, \beta] \), we require:

\[
f(n) \leq c' + \alpha \cdot n \log_2 n.
\]

This means:

\[
\alpha \geq \frac{\beta \max_{n=2} \frac{f(n) - c'}{n \cdot \log_2 n}}{n \cdot \log_2 n} \tag{15}
\]

**Inductive case.** Assuming \( f(n) \leq c' + \alpha \cdot n \log_2 n \) for all \( n \leq t - 1 \) where \( t \geq \beta + 1 \geq 2 \), we want to prove \( f(t) \leq c' + \alpha \cdot t \log_2 t \).

\[
f(t) \leq f(\lceil t/3 \rceil) + f(\lceil 2t/3 \rceil) + c \cdot t
\]

(by inductive assumption) \leq c' + \alpha \lceil t/3 \rceil \log_2 \lceil t/3 \rceil + c' + \alpha \lceil 2t/3 \rceil \log_2 \lceil 2t/3 \rceil + ct

For all \( t \geq 2 \), we have \( \lceil t/3 \rceil \leq t/2 \) and \( \lceil 2t/3 \rceil \leq t \). Furthermore, for any real number \( x \), \( \lceil x \rceil < x + 1 \).

Hence:

\[
\begin{align*}
f(t) & \leq 2c' + \alpha(t/3 + 1) \log_2 (t/2) + \alpha(2t/3 + 1) \log_2 t + ct \\
& = 2c' + \alpha(t/3 + 1)((\log_2 t) - 1) + \alpha(2t/3 + 1) \log_2 t + ct \\
& = 2c' + \alpha t \log_2 t - t(\alpha/3 - c) - \alpha + 2\alpha \log_2 t
\end{align*}
\]

To complete the inductive argument, we want:

\[
\begin{align*}
2c' + \alpha t \log_2 t - t(\alpha/3 - c) - \alpha + 2\alpha \log_2 t & \leq c' + \alpha t \log_2 t \\
\iff \alpha(t/3 - 2 \log_2 t + 1) & \geq ct + c' \tag{16}
\end{align*}
\]

We consider

\[
\beta \geq 128 \tag{17}
\]
under which $t \geq \beta + 1 \geq 129$ and, hence, $t/6 > 2\log_2 t$. Equipped with this, we get from (16):

$$\alpha \geq \frac{ct + c'}{t/3 - 2\log_2 t + 1}$$

$$\left\Rightarrow \begin{array}{c}
\alpha \geq \frac{2\max\{c, c'\} \cdot t}{t/3 - 2\log_2 t} \\
\alpha \geq \frac{2\max\{c, c'\} \cdot t}{t/3 - t/6} \\
= 12 \max\{c, c'\}
\end{array} \right.$$

(18)

Therefore, by choosing any $\beta$ satisfying (17) and any $\alpha$ satisfying (15) and (18), we have a working argument to show that $f(n) \leq c' + \alpha \cdot n \log_2 n$. 