1 Hierarchy

$$O(1) \leqslant O(log(n)) \leqslant O(n^c)$$
$$\leqslant O(n) \leqslant O(n^2)$$
$$\leqslant O(n^c) \leqslant O(c^n)$$

2 Big-O

We say that $f(n)$ grows asymptotically no faster than $g(n)$ if there is a constant $c_1 > 0$ such that $f(n) \leq c_1 \cdot g(n)$ and holds for all n at least a constant c_2. This is denoted by $f(n) = O(g(n))$.
$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$ for some constant c

2.1 Example

$1000 \log_2 n = O(n)$,
$n \neq O(10000 \log_2 n)$
$\log_{b_1} n = O(\log_{b_2} n)$ for any constants $b_1 > 1$ and $b_2 > 1$. Therefore $f(n) = 2+6\log_2 n$ can be represented; $f(n) = O(\log n)$

3 Big-Ω

If $g(n) = O(f(n))$, then $f(n) = \Omega(g(n))$ to indicate that $f(n)$ grows asymptotically no slower than $g(n)$. We say that $f(n)$ grows asymptotically no slower than $g(n)$ if $c_1 > 0$ such $f(n) \geq c_1 \cdot g(n)$ for $n > c_2$; denoted by $f(n) = \Omega(g(n))$

4 Big-Θ

If $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, then $f(n) = \Theta(g(n))$ to indicate that $f(n)$ grows asymptotically as fast as $g(n)$

When using 'Direction 1: Constant Finding' setting c_1, always set it to match the coefficent on the LHS so that you can cancel. When trying to get a contradiction, try and isolate an $x \cdot c_1$ on the RHS, where $x \in \mathbb{Z}$, such that an expression that contains n is $\leqslant x \cdot c_1$
Make judicious use of the *max* function when adding functions together If $f_1(n) + f_2(n) \leqslant c_1 \cdot g_1(n) + c_1' \cdot g_2(n) \leqslant max\{c_1, c_1'\} \cdot (g_1(n) + g_2(n))$, for all $n \geqslant max\{c_2, c_2'\}$.

5 The Master Theorem

5.1 Theorem 1

$n + \frac{n}{c} + \frac{n}{c^2} + \ldots + \frac{n}{c^h} = O(n)$

5.2 Theorem 2

Let $f(n)$ be a function that returns a positive value for every integer $n > 0$. We know:

$$f(1) \leqslant c_1$$
$$f(n) \leqslant \alpha \cdot f(\lceil n/\beta \rceil) + c_2 \cdot n^\gamma \text{ for } n \geqslant 2$$

where $\alpha, \beta, \gamma, c_1$ and c_2 are positive constants. Then:

- If $log_b\alpha < \gamma$ then $f(n) = O(n^\gamma)$
- If $log_b\alpha = \gamma$ then $f(n) = O(n^\gamma \cdot log(n))$
- If $log_b\alpha > \gamma$ then $f(n) = O(n^{log_\beta(a)})$