Worst-case Time Complexity: Proving Asymptotic Bounds.

Let $t(x)$ be the number of steps taken by algorithm $\mathcal{A}$ on input x.
Let $T(n)$ be the *worst-case* time complexity of algorithm $\mathcal{A}$:

$$T(n) = \max_{\text{all inputs } x \text{ of size } n} t(x) \;=\; \max\,\{t(x) : x \text{ is an input of size } n\}$$

1. To prove that $T(n)$ is $O(g(n))$, one must show that there is a constant $c > 0$, and an input size $n_0 > 0$, such that for all $n \geq n_0$:

$$\begin{array}{ll} & T(n) \leq c \cdot g(n) \\ \Leftrightarrow & \max\,\{t(x) : x \text{ is an input of size } n\} \leq c \cdot g(n) \\ \Leftrightarrow & \text{For } \textit{every} \text{ input } x \text{ of size } n,\ t(x) \leq c \cdot g(n) \\ \Leftrightarrow & \text{For } \underline{\textit{every} \text{ input of size } n,\ \mathcal{A} \text{ takes } \textit{at most } c \cdot g(n)} \text{ steps} \end{array}$$

2. To prove that $T(n)$ is $\Omega(g(n))$, one must show that there is a constant $c > 0$, and an input size $n_0 > 0$, such that for all $n \geq n_0$:

$$\begin{array}{ll} & T(n) \geq c \cdot g(n) \\ \Leftrightarrow & \max\,\{t(x) : x \text{ is an input of size } n\} \geq c \cdot g(n) \\ \Leftrightarrow & \text{For } \textit{some} \text{ input } x \text{ of size } n,\ t(x) \geq c \cdot g(n) \\ \Leftrightarrow & \underline{\text{For } \textit{some} \text{ input of size } n,\ \mathcal{A} \text{ takes } \textit{at least } c \cdot g(n)} \text{ steps} \end{array}$$

In summary:

Let $T(n)$ be the *worst-case* time complexity of algorithm $\mathcal{A}$.

1. $T(n)$ is $O(g(n))$ iff $\exists c > 0$, $\exists n_0 > 0$, such that $\forall n \geq n_0$:
   for *every* input of size n, $\mathcal{A}$ takes *at most* $c \cdot g(n)$ steps.
2. $T(n)$ is $\Omega(g(n))$ iff $\exists c > 0$, $\exists n_0 > 0$, such that $\forall n \geq n_0$:
   for *some* input of size n, $\mathcal{A}$ takes *at least* $c \cdot g(n)$ steps.
3. $T(n)$ is $\Theta(g(n))$ iff $T(n)$ is $O(g(n))$ and $T(n)$ is $\Omega(g(n))$.