نمادهای مجانبی :

O : این فرمول برای مقایسه رشد دو تابع در مقادیر بزرگ استفاده می‌شود. در این فرمول $T(n)$ و $g(n)$ دو تابع هستند که رشد آنها در مقادیر بزرگ مورد بررسی قرار می‌گیرد. c یک ثابت مثبت است و n_0 یک عدد مثبت است. فرمول بیان می‌کند که برای هر عدد n بزرگتر از n_0، مقدار $T(n)$ کمتر یا مساوی c ضرب در مقدار $g(n)$ است. به عبارت دیگر رشد $T(n)$ به صورت نمایی کمتر یا مساوی رشد $g(n)$ است. این فرمول برای تحلیل عملکرد الگوریتم‌ها و مقایسه عملکرد دو الگوریتم در مقادیر بزرگ استفاده می‌شود.

$$\begin{cases} T(n) \in O(g(n)) \iff \exists\, n_0,\ c>0 : \forall n \geq n_0,\ T(n) \leq c \cdot g(n) \\ T(n) \in O(g(n)) \iff \lim\limits_{n\to\infty} \dfrac{T(n)}{g(n)} = \begin{cases} c \ \text{عدد ثابت} \\ 0 \end{cases} \end{cases}$$

Ω : امگای بزرگ، پایین‌ترین حد عملکرد زمانی یک الگوریتم را مشخص می‌کند. به عبارت دیگر، امگای بزرگ نشان می‌دهد که برای ورودی‌های بزرگ، الگوریتم باید حداقل به اندازه‌ای زمان ببرد که توسط تابع $T(n)$ نشان داده شده است. در این فرمول، $g(n)$ تابعی است که پیچیدگی زمانی الگوریتم را توصیف می‌کند و c یک ثابت مثبت است. اگر $\Omega(g(n)=T(n)$، آنگاه برای هر ثابت مثبت $c*\ g(n) \leq T(n)$. به عبارت دیگر برای هر مقدار ثابت c، زمان اجرای الگوریتم برای ورودی‌های بزرگ، حداقل c برابر با $g(n)$ خواهد بود.

$$\begin{cases} T(n) \in \Omega(g(n)) \iff \exists\, n_0,\ c>0 : \forall n \geq n_0,\ T(n) \geq c \cdot g(n) \\ T(n) \in \Omega(g(n) \iff \lim\limits_{n\to\infty} \dfrac{T(n)}{g(n)} = \begin{cases} c \\ \infty \end{cases} \end{cases}$$

o : به طور ساده اگر تابع $T(n)$ در مقیاس بزرگ، کوچکتر یا برابر با تابع $g(n)$ باشد، می‌گوییم که $T(n) = o(g(n))$ است. به عبارت دیگر رشد $T(n)$ به صورت نمایی کمتر از رشد $g(n)$ است. این فرمول برای تحلیل عملکرد الگوریتم‌ها و مقایسه عملکرد دو الگوریتم در مقادیر بزرگ استفاده می‌شود.

$$T(n) = o(g(n)) \iff \forall\, c>0,\ \exists\, n_0>0 : \forall n \geq n_0,\ T(n) \leq c \cdot g(n)$$

Ω : تابع $\Omega(g(n))$ مجموعه‌ای از تمام توابع $T(n)$ را شامل می‌شود که برای مقادیر کافی بزرگ n، $T(n)$ از $c.g(n)$ بزرگتر یا برابر باشد. به عبارت دیگر $T(n)$ به طور مجانبی حداقل به اندازه $g(n)$ رشد می‌کند.

$$T(n) \in \Omega(g(n)) \Leftrightarrow \forall c>0, \exists n_0>0 : \forall n \geqslant n_0 , c.g(n) \leqslant T(n)$$

Θ : این فرمول پیچیدگی زمانی الگوریتم است. در این فرمول c_1 و c_2 مقادیر ثابتی هستند که به الگوریتم و داده های ورودی آن وابسته هستند. n نیز اندازه ورودی الگوریتم است.

$$\begin{cases} T(n) \in \Theta(g(n)) \Leftrightarrow \exists n_0 , c_1 , c_2 > 0 : \forall n \geqslant n_0 , c_1.g(n) \leqslant T(n) \leqslant c_2.g(n) \\ T(n) \in \Theta(g(n)) \Leftrightarrow \lim_{n \to \infty} \frac{T(n)}{g(n)} = c \end{cases}$$