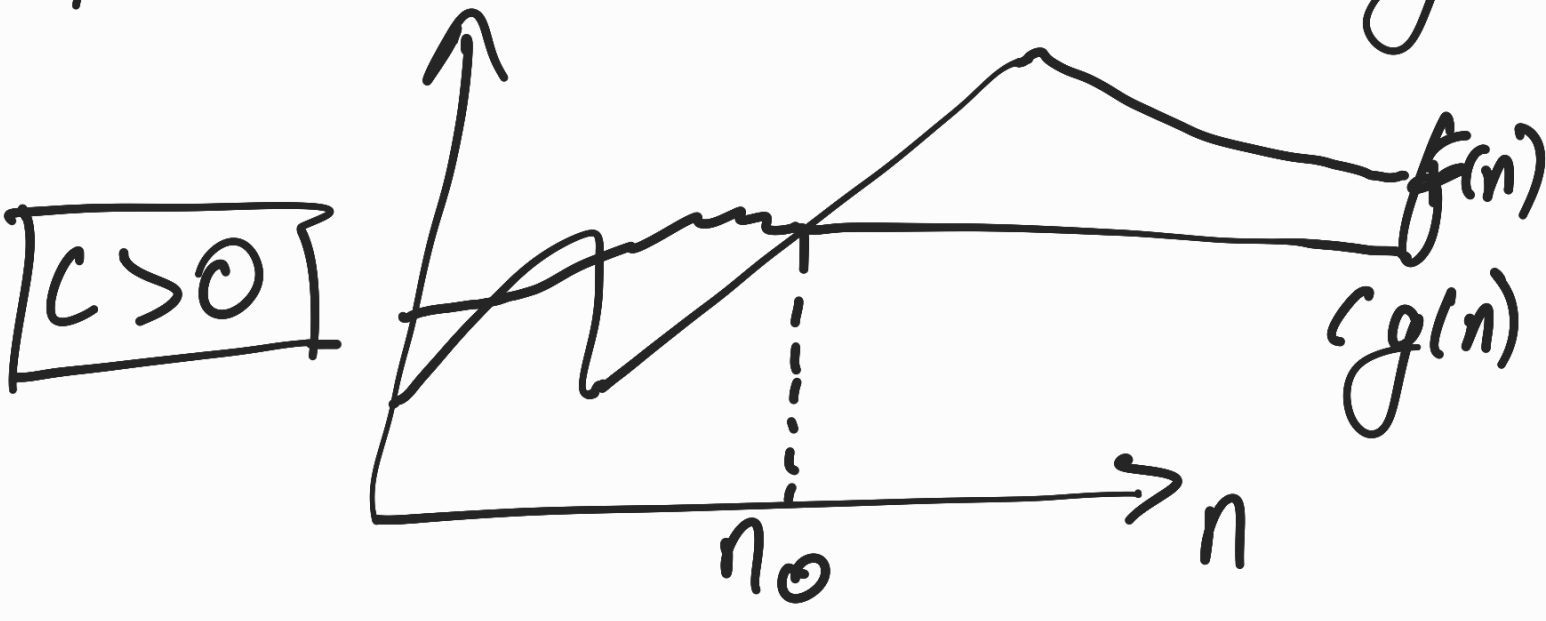
Asymtotic Notations

There are mainly three types of asymtotic Notation

- Omega Notation
- Big-O Notation
- Theta Notation

(i) Omega Notation (Ω-Notation)

⇒ Omega Notation represents the lower bound of the running time of an algorithm. Thus, it provides best time complexity.

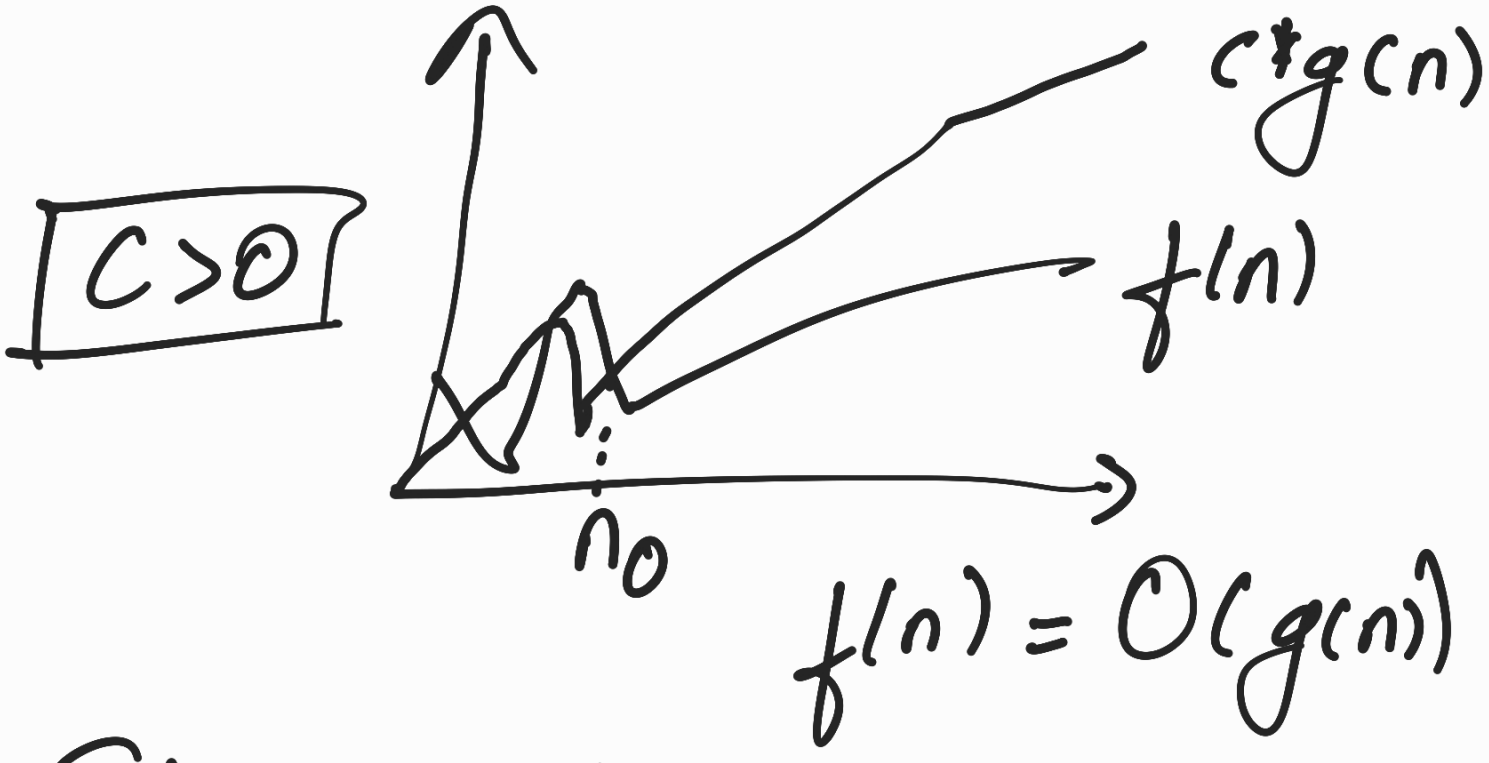


$f(n) = \text{Omega}(g(n))$

Let g & f be the set of natural numbers to itself. The function is said to be $\Omega(g)$. If there is a constant $c > 0$ and a natural number n_0 such that $c * g(n) \leq f(n)$ for all $n \geq n_0$.

(ii) Big-O Notation (O-Notation):

⇒ Big-O notation represents the upper bound of the running time of an algorithm. Therefore, it gives the worst case time complexity of an algorithm.



$f(n) = O(g(n))$

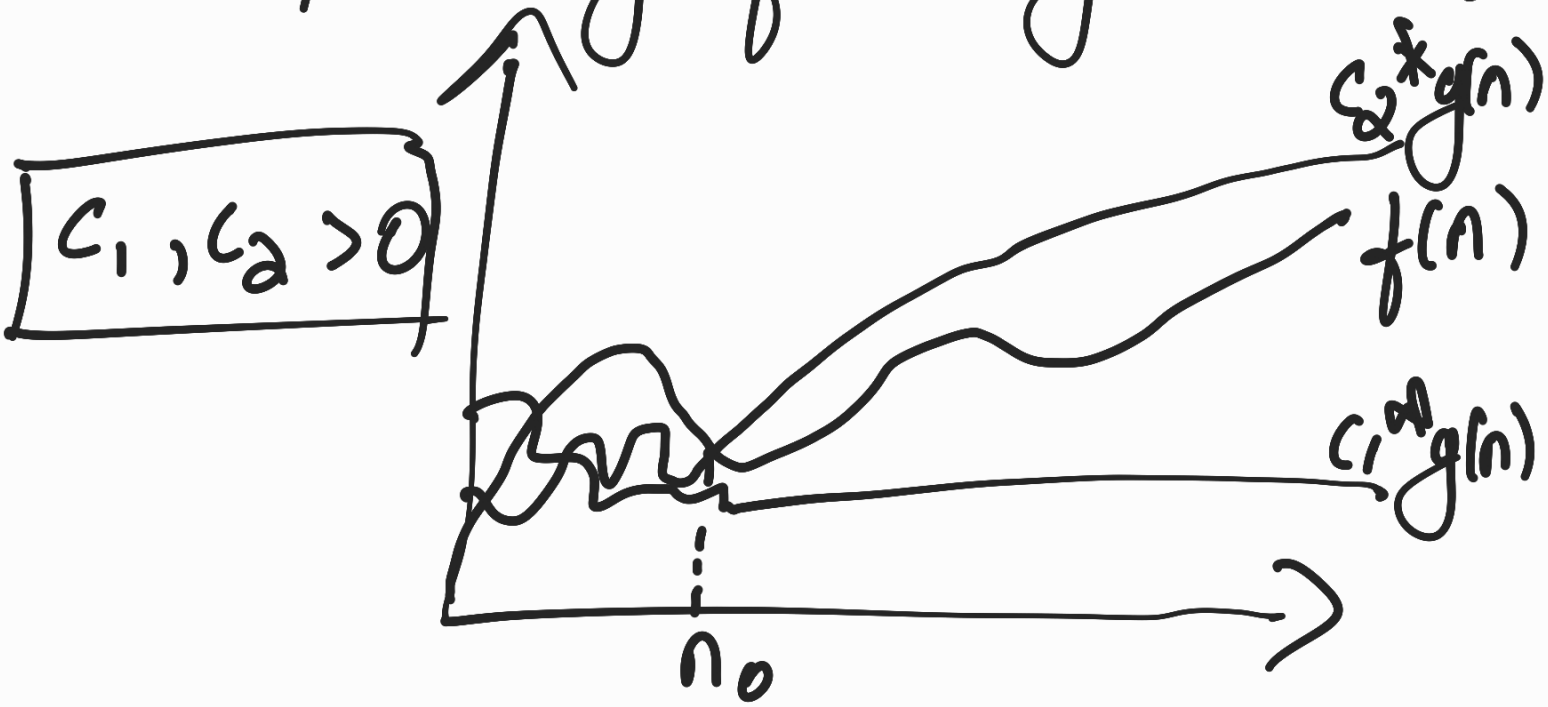
If $f(n)$ is the runtime of an algorithm, $f(n)$ is $O(g(n))$ if there exists a positive constant c and n_0 such that

$0 \leq f(n) \leq c * g(n)$ for all $n \geq n_0$

This, is the most widely used notation as when you are in a organization you have to consider the worst case scenario.

(iii) Theta Notation (Θ-notation):

⇒ Theta Notation encloses the function from above & below. Since it represents the lower & upper bound of the running time of an algorithm. It is used for analyzing the average-case complexity of an algorithm.



$f(n) = \Theta(g(n))$

Let g & f be the function from the set of natural numbers to itself. The function f is said to be $O(g)$. If there are constants $c_1, c_2 > 0$ & a natural number n_0 such that

$$c_1 * g(n) \leq f(n) \leq c_2 * g(n) \text{ for all } n \geq n_0$$