Asymptotic Analysis :-

The time required by an algorithm comes under three types :

Worst case :- It defines the input for which the algorithm takes a huge time.

Average case :- It takes average time for the program execution.

Best case :- It defines the input for which the algorithm takes the lowest time.

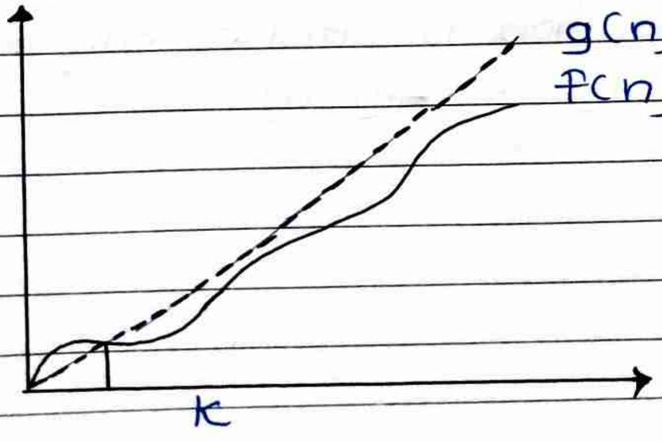
Asymptotic Notations :-

The commonly used asymptotic notations used for calculating the running time complexity of an algorithm is given below :

1). Big oh notation (O) :-

This measures the performance of an algorithm by simply providing the order of growth of the function.

This notation provides an upper bound on a function which ensures that function never grows faster than the upper bound.



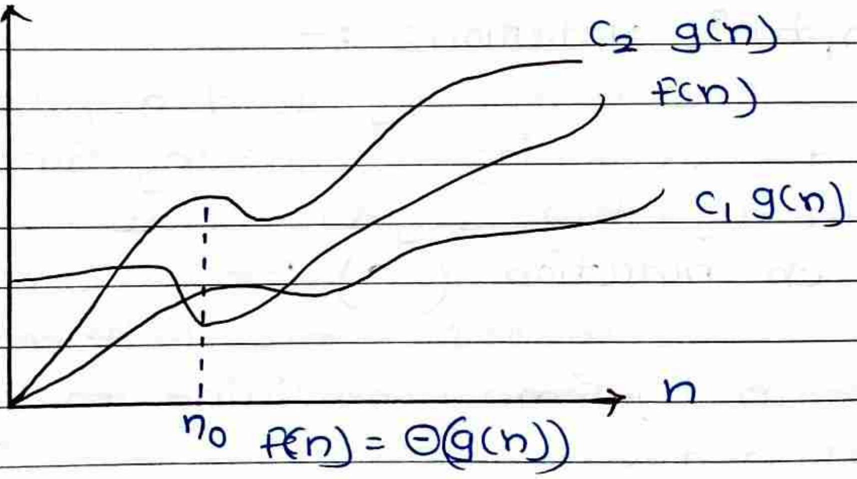
Example :- If f(n) and g(n) are two functions defined for positive integer,

then $f(n) = O\, g(n)$ as f(n) is big oh of g(n) or f(n) is on order of g(n)) if there exists constants c and no such that :

$$f(n) \leq c \cdot g(n) \text{ for all } n \geq n_0$$

2>. Omega Notation (Ω) :-

It basically describes best case scenario which is opposite to big o notation. It is the formal way to represent lower bound of an algorithm's running time.



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

Example :- let f(n) and g(n) be functions of n where n is steps required to execute program:

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

The above condition is satisfied only if when:

$c1.\, g(n) <= f(n) <= c2.g(n)$

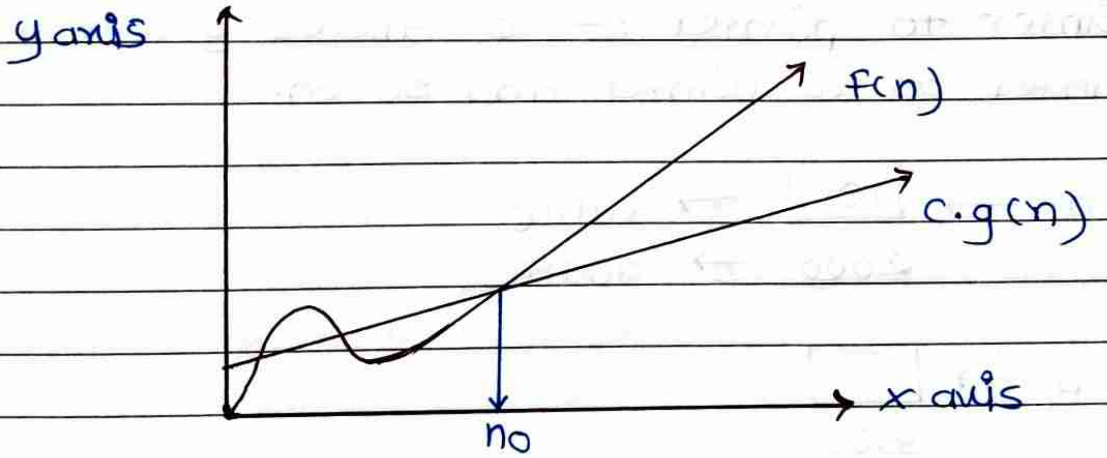
2). omega Notation (Ω)

It basically describes best-case scenario which is opposite to big-o notation It is formal way to represent lower bound to an algorithm's running time. It measures the best amount of time an algorithm can possibly take to complete or best case time complexity.

Example :- If f(n) and g(n) are two functions defined for positive integers,

then $f(n) = \Omega g(n)$ as f(n) is omega of g(n) or f(n) is on the order of g(n) if there exists constants c and no such that:

$f(n) >= c \cdot g(n)$ for all $n \geq n_0$ and $c > 0$



3). Theta Notation (Θ)

The theta notation mainly describes average case scenarios.

It represents realistic time complexity of an algorithm. Big theta is mainly used when the value of worst-case and best case is same.