

Python - Algorithm Types

The efficiency and accuracy of algorithms have to be analysed to compare them and choose a specific algorithm for certain scenarios. The process of making this analysis is called Asymptotic analysis. It refers to computing the running time of any operation in mathematical units of computation.

For example, the running time of one operation is computed as $f(n)$ and may be for another operation it is computed as $g(n^2)$. This means the first operation running time will increase linearly with the increase in n and the running time of the second operation will increase exponentially when n increases. Similarly, the running time of both operations will be nearly the same if n is significantly small.

Usually, the time required by an algorithm falls under three types –

- **Best Case** – Minimum time required for program execution.
- **Average Case** – Average time required for program execution.
- **Worst Case** – Maximum time required for program execution.

Asymptotic Notations

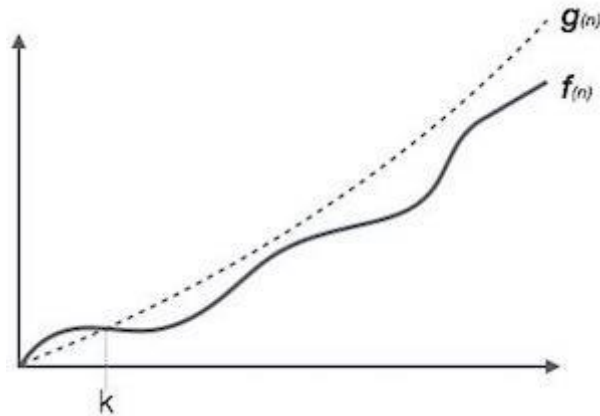
The commonly used asymptotic notations to calculate the running time complexity of an algorithm.

- O Notation

- Ω Notation
- θ Notation

Big Oh Notation, O

The notation $O(n)$ is the formal way to express the upper bound of an algorithm's running time. It measures the worst case time complexity or the longest amount of time an algorithm can possibly take to complete.

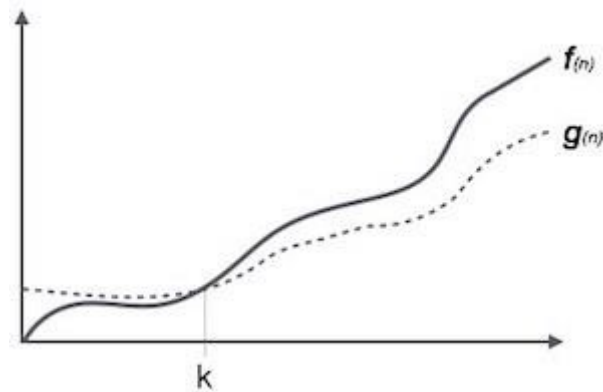


For example, for a function $f(n)$

$$O(f(n)) = \{ g(n) : \text{there exists } c > 0 \text{ and } n_0 \text{ such that } f(n) \leq c \cdot g(n) \text{ for all } n > n_0. \}$$

Omega Notation, Ω

The notation $\Omega(n)$ is the formal way to express the lower bound of an algorithm's running time. It measures the best case time complexity or the best amount of time an algorithm can possibly take to complete.

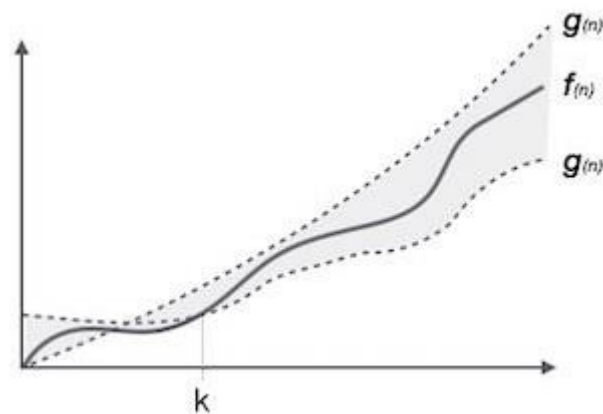


For example, for a function $f(n)$

$$\Omega(f(n)) \geq \{ g(n) : \text{there exists } c > 0 \text{ and } n_0 \text{ such that } g(n) \leq c \cdot f(n) \text{ for all } n > n_0. \}$$

Theta Notation, θ

The notation $\theta(n)$ is the formal way to express both the lower bound and the upper bound of an algorithm's running time. It is represented as follows –



$$\theta(f(n)) = \{ g(n) \text{ if and only if } g(n) = O(f(n)) \text{ and } g(n) = \Omega(f(n)) \text{ for all } n > n_0. \}$$

Common Asymptotic Notations

A list of some common asymptotic notations is mentioned below –

constant	–	$O(1)$
logarithmic	–	$O(\log n)$
linear	–	$O(n)$
$n \log n$	–	$O(n \log n)$
quadratic	–	$O(n^2)$
cubic	–	$O(n^3)$
polynomial	–	$n^{O(1)}$
exponential	–	$2^{O(n)}$

 **Print Page**
