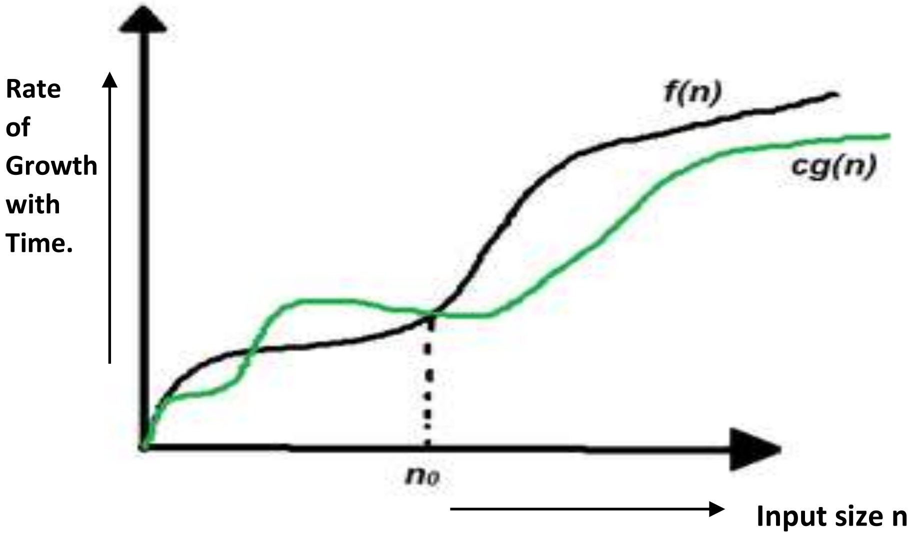
# BIG OMEGA NOTATION

The lower bound of an algorithm is given by the big-omega $(\Omega)$ notation.




**DEFINITION**: A function $f(n)$ is said to be in $\Omega\big(g(n)\big)$, denoted $f(n)\ \varepsilon\ \Omega\big(g(n)\big)$, *if $f(n)$ is bounded below by some positive constant multiple of $g(n)$ for all large n , i.e., if there exist some positive constant c and some nonnegative integer $n_0$ such that*:

$$f(n) \geq c\ \times g(n)\ for\ all\ n \geq n_0$$

# ILLUSTRATION OF THE DEFINITION

- Let $f$ $and$ $g$ be two functions that map a set of natural numbers to a set of positive real numbers , that is $f: \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$ .
- Let $\Omega(g)$ be the set of all those functions that have a similar rate of growth.
- The relation $f(n) = \Omega(g(n))$ holds good if there exist two positive constants $c$ $and$ $n_0$ such that $f(n) \geq c \times g(n)$.
- Thus, the function $f(n)$ is said to be in $\Omega(g(n))$ , which can be represented as $f(n) \epsilon\ \Omega(\mathrm{g}(\mathrm{n}))$.
- This notation implies that $f(n)$ grows at a faster rate than a constant time $g(n)$ for a sufficiently large $n$ .

***The "omega notation" is used when the lower bound of a polynomial is to be found.***

## THE NEED OF BIG OMEGA ($\Omega$) NOTATION:

- The notation is helpful in finding out the minimum amount of resources, an algorithm requires, in order to run.
- Finding out the minimum amount of resources is important as this time complexity can help us to schedule the task accordingly.
- It is also helpful to compare the best suited algorithm amongst the set of algorithms, if more than one algorithm can accomplish a given task.

Hence:

$\boldsymbol{f(n) = \Omega(g(n)), if\ f(n) \geq c\ \times g(n), n \geq n_0, where\ c\ and\ n_0\ are}$ **constants.**

i.e.

$$\boldsymbol{\Omega(g(n)) = \{f(n)\!: there\ exists\ positive\ constants\ c\ and\ n_0}$$

$$\boldsymbol{such\ that\ 0 \leq c \times g(n) \leq f(n)\ for\ all\ n \geq n_0\}}$$

- $\boldsymbol{g(n)\ is\ an\ asymptotic\ tight\ lower\ bound\ for\ f(n)}$.
- Hence the Big-Omega notation gives the tighter lower bound for the given algorithm.
- Our objective is to give the largest rate of growth $\boldsymbol{g(n)}$ which is less than or equal to the given algorithm's rate of growth $\boldsymbol{f(n)}$ .

***********