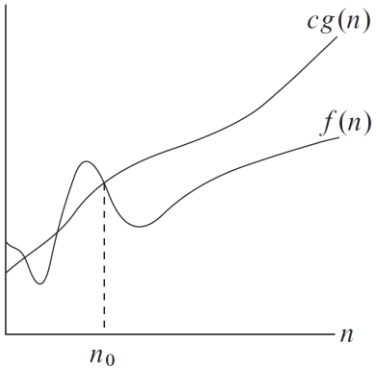
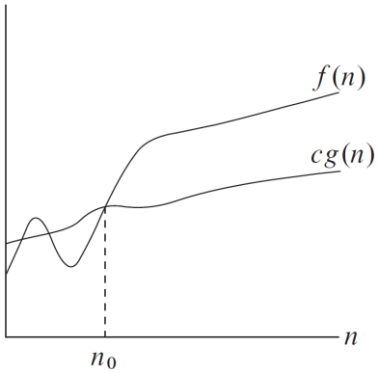
Asymptotic notations are the mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value. There are mainly three asymptotic notations:

- Big-O notation: It represents the upper bound of the running time of an algorithm. The following expression can be described as a function *f(n)* belongs to the set *O(g(n))* if there exists a positive constant *c* such that it lies between *0* and *cg(n)*, for sufficiently large *n*.



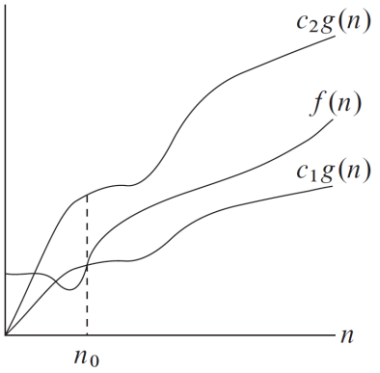
$$O(g(n)) = \{f(n): \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0\}$$

- Omega notation: It represents the lower bound of the running time of an algorithm. The following expression can be described as a function *f(n)* belongs to the set *Ω(g(n))* if there exists a positive constant *c* such that it lies above *cg(n)*, for sufficiently large *n*.



$$\Omega(g(n)) = \{f(n): \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\}$$

- Theta notation: It represents the upper and the lower bound of the running time of an algorithm. The following expression can be described as a function *f(n)* belongs to the set *Θ(g(n))* if there exist positive constants c_1 and c_2 such that it can be sandwiched between $c_1g(n)$ and $c_2g(n)$, for sufficiently large n.



$$\Theta(g(n)) = \{f(n): \text{there exist positive constants } c_1, c_2 \text{ and } n_0 \text{ such that } 0 \le c_1g(n) \le f(n) \le c_2g(n) \text{ for all } n \ge n_0\}$$