$O(n)$

$f(n) = O(g(n))$ | $f(n) = \Omega(g(n))$

$f(n) <= c * g(n)$ | $c * g(n) <= f(n)$.

Lower bound → $f(n) = 100n + 5$.

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

$\Rightarrow c \cdot g(n) <= f(n)$.

$c \,g(n) <= 100n + 5$.

c → Trial && Error.

$g(n) = n$,

$cn <= 100n + 5$.

c
105
210

$c = 105$,

$105n <= 100n + 5$

$5n <= 5$

$\Rightarrow n <= 1$

$n >= n_0$

$cn <= 100n + 5$

$c = 1$, $n <= 100n + 5$

+ve +ve

$n >= 0 \rightarrow n_0 = 0$

$c = (1, 100)$

$c=1, \ n <= 100n+5$ $\quad n>=0 \rightarrow n_0=0$

+ve $\quad$ +ve

$c=2 \quad 2n <= 100n+5 \ , \ n>=0 \rightarrow n_0=0 \quad 0 <= 5$

$c=100, \quad 100n <= 100n+5 \Rightarrow 0<=5,$

$c>0, \quad n_0>=0 \quad , n>=n_0$

- Transitivity: $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$. Valid for O and Ω as well.
- Reflexivity: $f(n) = \Theta(f(n))$. Valid for O and Ω.
- Symmetry: $f(n) = \Theta(g(n))$ if and only if $g(n) = \Theta(f(n))$.
- Transpose symmetry: $f(n) = O(g(n))$ if and only if $g(n) = \Omega(f(n))$.
- If $f(n)$ is in $O(kg(n))$ for any constant $k > 0$, then $f(n)$ is in $O(g(n))$.
- If $f_1(n)$ is in $O(g_1(n))$ and $f_2(n)$ is in $O(g_2(n))$, then $(f_1 + f_2)(n)$ is in $O(\max(g_1(n)), (g_1(n)))$.
- If $f_1(n)$ is in $O(g_1(n))$ and $f_2(n)$ is in $O(g_2(n))$ then $f_1(n)\, f_2(n)$ is in $O(g_1(n)\, g_1(n))$.

$f(n) = O(\underbrace{k \times g(n)}) \ , k>0.$

$\Rightarrow f(n) <= \underline{c} \cdot \underline{k} \cdot g(n),$

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

$\Rightarrow f(n) <= c \cdot g(n).$

$\Rightarrow f(n) <= K \cdot g(n)$.

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

Given,

$\Rightarrow f1(n) = O(g_1(n))$; $f2(n) = O(g_2(n))$

$\Rightarrow f1(n) <= c_1 \cdot g_1(n)$, $f2(n) <= c_2 \cdot g_2(n)$

Multiply

$f1(n) \cdot f2(n) <= c_1 g_1(n) \cdot c_2 g_2(n)$

$f_1(n) \cdot f2(n) <= C \times g_1(n) \cdot g_2(n)$

$\Rightarrow f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$

To prove,

$f_1(n) \cdot f_2(n) = O(g_1(n) g_2(n))$

$f_1(n) \cdot f_2(n) <= C \cdot g_1(n) \cdot g_2(n)$

for (i = 1 ; i <= n ; i++) TC $\rightarrow$?? O(n).

```
for(i=1;i<=n;i++)      TC = O(n)
        print

for(i=1 i<=n ;i++){ → n
    for(j=1;j<=n;j++){ → n      TC = O(n×n) = O(n^2)
            print();
    }
}
```

```
for(i=1;i<=n;i++){ → O(n)
        print()
}
for(j=1;j<=n;j++){ → O(n)
        print()
}
```

}

```
for (i=1; i<=n; i=i*2){
    print();
}
```

```
for (i=n; i>=1; i=i/2){
    print();
}
```

TC → O(log n)

i → Linearly increase / decrease

i → Drastically increase / decrease

i = 1, 2, 4, 8, 16, ..., ~~value~~ n

$2^0 \quad 2^1 \quad 2^2 \quad 2^3 \quad 2^4 \quad \dots \quad 2^k = 2^{k+1}$

k^{th} step $\qquad (k+1)^{th}$ step

$\log_2(\text{value}) = \log_2(2^k) = k \qquad \log(n).$