Date.
Page No.

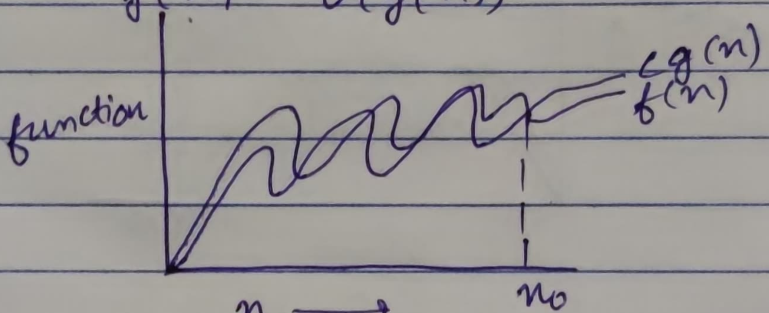
Tutorial-1

1. **Asymptotic Notations** - They 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.

Different asymptotic notations -

i) Big $O(n)$

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



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

iff $\quad f(n) \le c\,g(n)$

$\forall\, n \ge n_0$

for some constant, $c > 0$

$g(n)$ is "tight" upper bound of $f(n)$.

ex. $f(n) = n^2 + n$

$g(n) = n^3$

$n^2 + n \le cn^3$

$n^2 + n = O(n^3)$

ii) Big omega (Ω)

Date.
Page No.

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

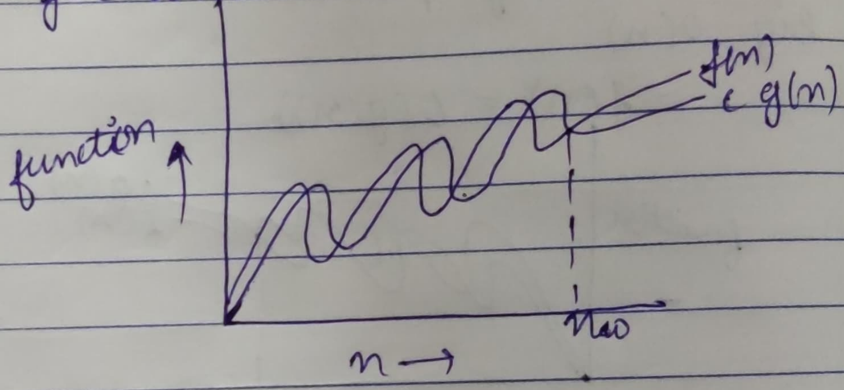
g(n) is "tight" lower bound of function f(n).

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

iff $f(n) \geq c\,g(n)$

$\forall\ n \geq n_0$

for some constant $c > 0$



ex.

$$f(n) = n^3 + 4n^2$$

$$g(n) = n^2$$

$$n^3 + 4n^2 = \Omega(n^2)$$

iii) Big Theta (Θ)

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

g(n) is both "tight" upper and "lower" bound of function f(n).

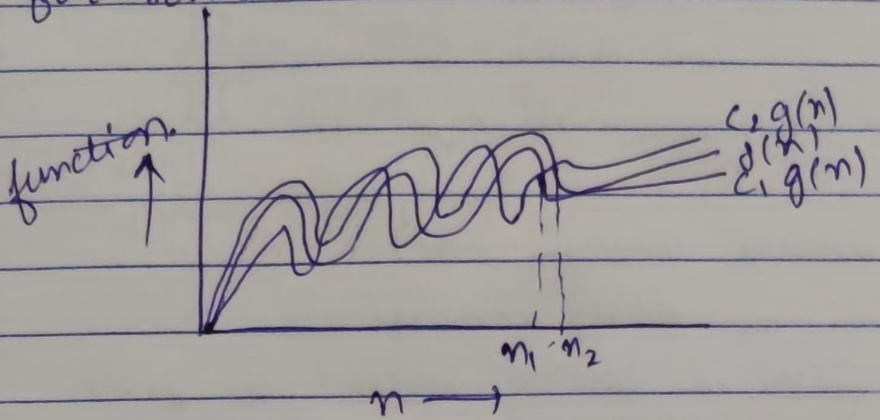
$$f(n) = \Theta(g(n))$$

iff

$$c_1\,g(n) \leq f(n) \leq c_2\,g(n)$$

$$\forall\ n \geq \max(n_1, n_2)$$

for some constant $c_1 > 0$ and $c_2 > 0$



Ex-

$3n+2 = o(n)$ as $3n+2 \geq 3n$ and

$3n+2 \leq 4(n)$ for n, $k_1 = 3$, $k_2 = 4$ & $n_0 = 2$

iv) Small o (o) -

$$f(n) = o(g(n))$$

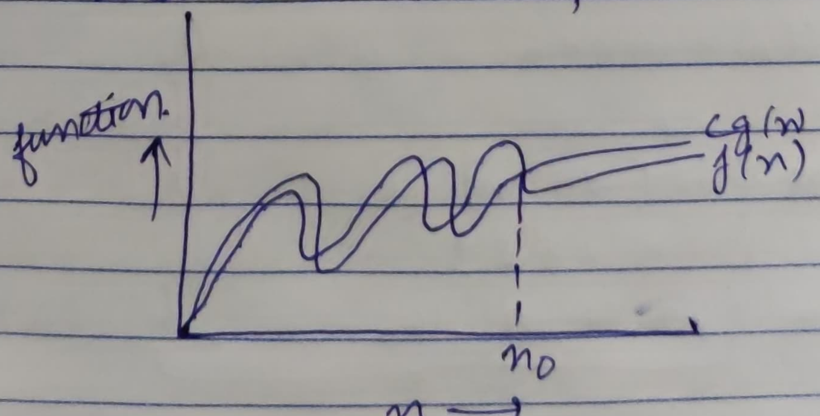
$g(n)$ is upper bound of function $f(n)$.

$$f(n) = o(g(n))$$

when $f(n) < c\,g(n)$

$\forall\ n > n_0$

and $\forall$ constants, $c > 0$



Date.
Page No.

Ex - $f(n) = n^2$

$g(n) = n^3$

$n^2 = O(n^3)$

v. Small omega (ω)

$f(n) \geq \omega(g(n))$.

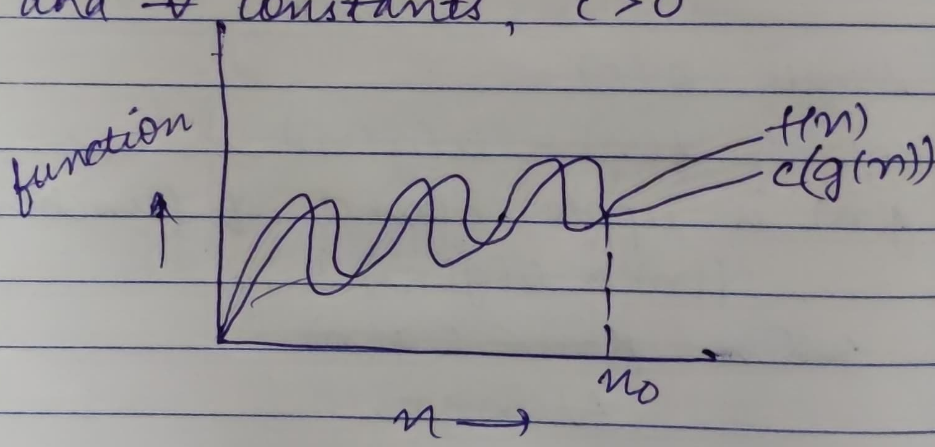
g(n) is lower bound of f(n).

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

when $f(n) > c\, g(n)$

$\forall\, n > n_0$

and $\forall$ constants, $c > 0$



$f(n) = 4n+6 \qquad g(n) = (1)$

2. for (i=1 to n)

{ i = i * 2 }

→ $i = \underbrace{1, 2, 4, 8, 16, \ldots, n}_{k}$ (G.P.)

$- O(k)$

$a = 1, \quad r = \frac{2}{1} = 2.$

Date.
Page No.

GP kth value $= t_k = ar^{k-1}$

$$n = 1 \times 2^{k-1}$$

$$n = \frac{2^k}{2}$$

$$2n = 2^k$$

$$\log(2n) = k \log 2$$

$$k = \log_2 2n$$

$$k = \log_2 2 + \log_2 n$$

$$k = 1 + \log n$$

Time comp $= O(1 + \log_2 n)$

$= O(\log_2 n)$

3. $T(n) = 3T(n-1)$ — ①

Let $n = n-1$

$T(n-1) = 3T(n-2)$ — ②

Put ② in ①

$T(n) = 3 \times 3T(n-2)$ — ③

Put $n = n-2$

$T(n-2) = 3T(n-3)$ — ④

Put ④ in ③

$T(n) = 3 \times 3 \times 3T(n-3)$ — ⑤

$T(n) = 3^n T(n-n)$

$= 3^n T(0)$

$= 3^n$

$= O(3^n)$

Date.
Page No.

4. $T(n) = 2T(n-1) - 1$

$= 2(2T(n-2) - 1) - 1$

$= 2^2(T(n-2)) - 2 - 1$

$= 2^3 T(n-3) - 2^2 - 2^1 - 2^0$

$\ldots$

$= 2^n T(n-n) - 2^{n-1} - 2^{n-2} - 2^{n-3} - \ldots - 2^2 - 2^1 - 2^0$

$= 2^n - 2^{n-1} - 2^{n-2} - 2^{n-3} \ldots - 2^2 - 2^1 - 2^0$

$= 2^n - (2^n - 1)$

$T(n) = 1$

5.
```
int i=1, S=1;
while (S<=n){
    i++; S= S+i;
    printf("#");
}
```

$S_i = S_{i-1} + i$

i is incrementing by one step.
S in incrementing by value of i
Following will be values after few iterations-

→ i=2, S=3 — 1st iteration
→ i=3, S=6 — 2nd iteration
→ i=4, S=10 — 3rd iteration

Let the value of n be k

Date.
Page No.

Values of $s = 1, 3, 6, 10, ----$
s represents a series of sum of first n natural numbers for $i = k$,

$$s = \frac{k(k+1)}{2} \text{ for stopping loop.}$$

$$\frac{k(k+1)}{2} > n \Rightarrow \frac{k^2+k}{2} > n$$

$$T(n) = O(\sqrt{n})$$

6.
```
void function (int n){
        int i, count = 0;
        for (i=1; i* i<=n; i++)
              count++;
}
```

$i = 1, 2, 3, --- n$

$i^2 = 1, 4, 9, --- n$

so $i^2 <= n$ or $i <= \sqrt{n}$

$$a_k = a + (k-1)d$$

$$a = 1, d = 1$$

$$a_k \leq \sqrt{n}$$

$$\sqrt{n} = 1 + (k-1) \cdot 1$$

$$\sqrt{n} = k$$

$$T(n) = O(\sqrt{n})$$

Date.
Page No.

7.
```
void function (int n){
    int i, j, k, count = 0;
    for ( i= n/2 ; i<= n; i++) {
        for (j=1; j<=n; j=j*2){
            for (k=1; k<=n; k=k*2){
                count++;
            }
        }
    }
}
```

$i = n/2$ $\quad$ $j = \log_2 n$ $\quad$ $k = \log_2 n$

$\left(\frac{n}{2}+1\right)$ times $\quad$ $\log_2 n$ $\quad$ $\log_2 n$

$$O(i*j*k) = O\left(\left(\frac{n}{2}+1\right) * \log_2 n * \log_2 n\right)$$

$$= O\left(\left(\frac{n}{2}+1\right) \times (\log n)^2\right)$$

$$T(n) = O(n(\log n)^2)$$

8.
```
function ( int n){
    if (n==1) return;
    for (i= 1 to n){
        for (j= 1 to n){
            print ("*");
        }
    function(n-3);
}
```

Date.
Page No.

$$T(n) = T(n-3) + n^2 \quad \text{—} \quad (1)$$

$$T(1) = 1 \quad \text{—} \quad (2)$$

put $n = n-3$ in (1)

$$T(n-3) = T(n-6) + (n-3)^2 \quad \text{—} \quad (3)$$

put (3) in (1)

$$T(n) = T(n-6) + (n-3)^2 + n^2 \quad \text{—} \quad (4)$$

put $n = n-6$ in (1)

$$T(n-6) = T(n-9) + (n-6)^2 \quad \text{—} \quad (5)$$

put (5) in (4)

$$T(n) = T(n-9) + (n-6)^2 + (n-3)^2 + n$$

generalising

$$T(n) = T(n-3k) + (n-3(k-1))^2 + (n-3(k-2))^2$$
$$\text{---} + n^2$$

let $n-3k = 1$

$$\frac{n-1}{3} = k$$

$$T(n) = T(1) + \left(n-3\left(\frac{n-1}{3}-1\right)\right)^2 +$$
$$\left(n-3\left(n-\frac{1}{3}\right)\right)^2 + \text{---} + n^2$$

$$T(n) = T(1) + [n-((n-1)-3]^2 + [n-(n-1-6)]^2$$
$$+ [n-(n-1-9)]^2 + \text{---} + n^2$$

$$T(n) = 1 + (3+1)^2 + (6+1)^2 + \text{---} + n^2$$

$$T(n) = 1^2 + 4^2 + 7^2 + \text{---} n^2$$

$$T(n) = n^2 + \text{---} 1$$

$$\boxed{T(n) = O(n^2)}$$

Date.
Page No.

9. void function (int n){
    for (i=1 to n) {
        for (j=1; j<=n; j=j+i){
            printf ("*");
        }
    }
}

for $i=1$, $j \Rightarrow n$ times

for $i=2$, $j = 1+3+5+ \cdots +n$

$$a_n = a + (k-1)d$$

$$a=1 \quad d=2$$

$$n = 1 + (k-1) \times 2$$

$$\frac{n-1}{2} = k-1$$

$$k = \frac{n-1}{2} + 1$$

$$\boxed{k = \frac{n+1}{2}}$$ no. of terms.

for $i=2$, $j \to \frac{n+1}{2}$ times

for $i=3$, $j = 1+4+7+ \cdots n$

$$n = 1 + (k-1) \times 3$$

$$\boxed{\frac{n-1}{3} + 1 = k}$$

for $i=3$, $j = \frac{n+2}{3}$ times

Date.
Page No.

generalising

for $i = n$, $j = \frac{n+k-1}{k}$ times

Time complexity is

$$\underbrace{n + \frac{n+1}{2} + \frac{n+2}{3} + \text{---} + \frac{n+k-1}{k}}_{n \text{ terms}}$$

general term $= \frac{n+k-1}{k}$

$$\sum_{k=1}^{n} \frac{n+k-1}{k} = \frac{\sum_{1}^{n} n + \sum_{1}^{n} k - \sum 1}{k}$$

$$\Rightarrow \frac{\frac{n(n+1)}{2} + nk - n}{k}$$

$$\Rightarrow \frac{n^2 + \frac{n}{2} + nk - n}{k}$$

$$T(n) = \frac{n^2 + \frac{n}{2} + nk - n}{k}$$

neglecting constant terms

$$\boxed{T(n) = O(n^2)}$$

Date.
Page No.

10. as given n^k & c^n
relation b/w n^k & c^n is
$$n^k = O(c^n)$$
as $n^k \le d c^n$
$\forall n \ge n_0$, d some constant $a > 0$
for $n_0 = 1$
$c = 2$
$\Rightarrow 1^k \le d_2$
$n_0 = 1$ & $c = 2$