TUTORIAL - I

1) **Big Oh** $f(n) = O(g(n))$; if $0 \leq f(n) \leq g(n)$, $\forall\, n \geq n_0$ for som const. $c > 0$

thu is tight upper bound of fn^n $f(n)$

if $g(n) = f(n)$ (but not $\leq$)

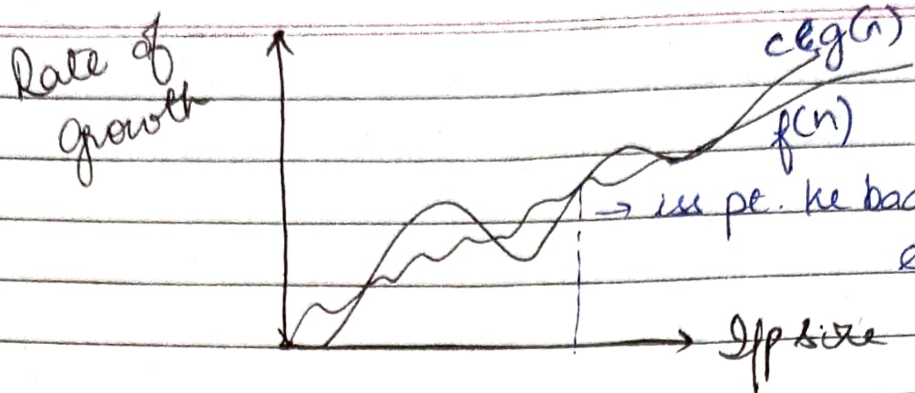
Rules - (i) constants are ignored if it comes as $+, -, *, /$

Ex → $A_1 - O(n^2 + 3n + 6) \rightarrow O(n^2 + n)$

$A_2 - O(2n^2 + 4n + 9) \rightarrow O(n^2 + n)$

(ii) lower order terms are ignored in $+, -$

Page ____ Date ____



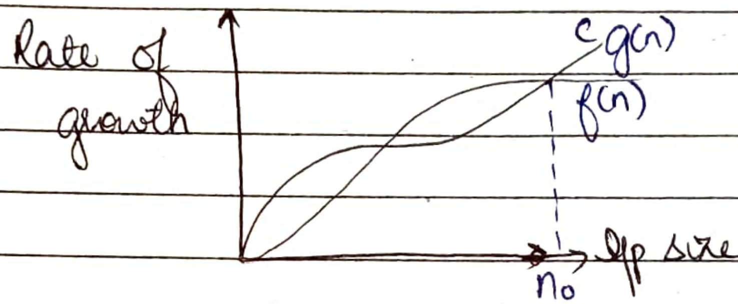
→ iss pt. ke baad ya toh zyada hoga ya equal.

3/3/22

ASYMPTOTIC NOTATNS

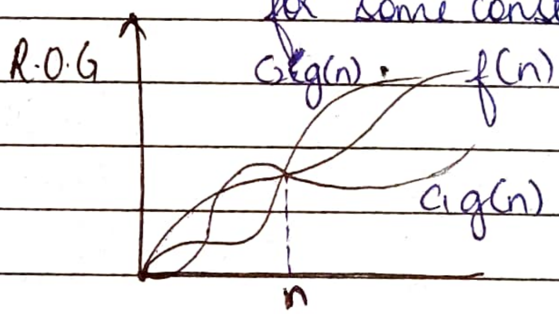
② SMALL OH (o)

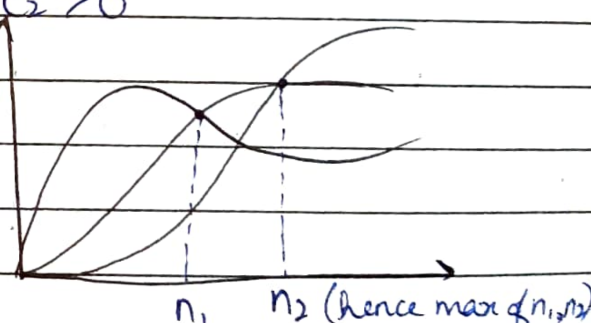
$f(n) = o\,g(n)$ if $f(n) < g(n)$
$\forall\, n > n_0$ & $\forall\, c > 0$



③ THETA (Θ)

$f(n) = \Theta(g(n))$ if $c_1 g(n) \le f(n) \le c_2 g(n)$
$\forall\, n \ge \max(n_1, n_2)$
for some consts. c_1 & $c_2 > 0$

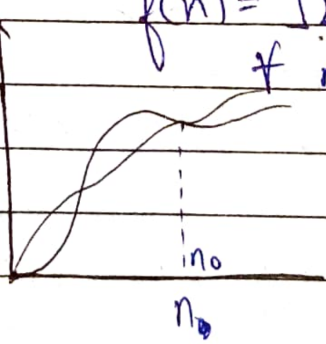




④ BIG OMEGA (Ω)

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

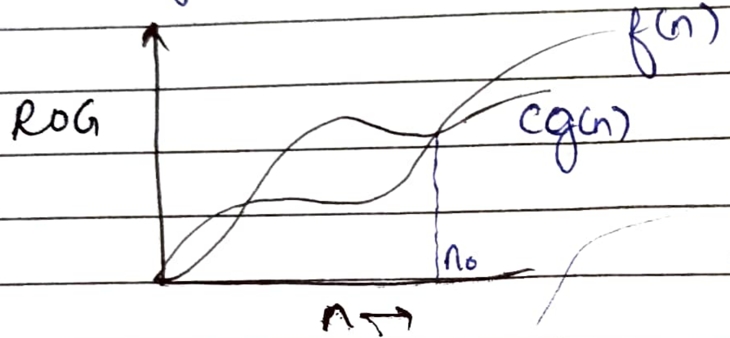
$f(n) = \Omega(g(n))$ if $f(n) \ge c\,g(n)$
$\forall\, n \ge n_0$ & some const. $c > 0$



Date

④ SMALL OMEGA (ω) gives lower bound

$f(n) = \omega(g(n))$ if $f(n) > c\,g(n)$ $\forall\, n > n_0$ & $\forall\, c > 0$



III) for $i = 1, 2, 4, 6, 8 \ldots$ n times

i.e series is a G.P.

So $a = 1$, $r = 2/1$; k^{th} val. of G.P

$t_k = ar^{k-1}$, $t_k = 1(2)^{k-1}$; $2n = 2^k$

$\log_n(2n) = k\log 2 \Rightarrow \log_2 2 + \log_2 n = k$

$\Rightarrow \log_2 n + 1 = k$ (neglecting '1')

So, time complexity $T(n) = O(\log_2 n)$

IV) $T(n) = \{2T(n-1) - 1$ if $n > 0$, otherwise $1\}$

$T(n) = 2T(n-1) - 1 \rightarrow$ (I) , put $n = n-1$

$T(n-1) = 2T(n-2) - 1 \rightarrow$ (II) , (II) in (I) :-

$T(n) = 2 \cdot 2T(n-1) - 2 - 1 \Rightarrow 4T(n-2) - 3 \rightarrow$ (III)

$T(n-2) = 2T(n-3) - 3 \rightarrow$ (IV) put in (III)

$T(n) = 4 \cdot 2T(n-3) - \overset{4-2-1}{\cancel{\;}} = 8T(n-3) - 15$

k^{th} term :- Let $n-k = 1 \Rightarrow k = n-1$

$T(n) = 2^{n-1}T(1) - 2^k(1/2 + 1/2^2 + \ldots + 1/2^k)$

$= 2^{n-1} - 2^{n-1}(1/2 + 1/2^2 + 1/2^{k-1})$

In Series in GP :- $a = 1/2$, $r = 1/2$

So, $T(n) = 2^{n-1}\left(1 - \left(\frac{1/2(1-(1/2)^{n-1})}{1-1/2}\right)\right)$

$= \frac{2^{n-1}}{\cancel{\;}}(1 - 1 + (1/2)^{n-1})) \Rightarrow \frac{2^{n-1}}{2^{n-1}} \Rightarrow T(n) = O(1)$

DATE

Q-6) Time complexity of:

↳ As $i^2 = n$

$i = \sqrt{n}$

$i = 1, 2, 3, 4, \ldots \sqrt{n}.$

$\sum_{i=1} 1+2+3+4+ \ldots + \sqrt{n}.$

$$T(n) = \frac{\sqrt{n} * (\sqrt{n}+1)}{2}$$

$$T(n) = \frac{n * \sqrt{n}}{2} \Rightarrow T(n) = O(n) \text{ Ans.}$$

Q-7) Since for $k = k^2$

$K = 1, 2, 4, 8 \ldots k.$

∴ series is in GP.

So $a = 1, r = 2$

$$\frac{a(r^n - 1)}{r-1} \Rightarrow \frac{1(2^k - 1)}{1}$$

$n = 2^k - 1 \Rightarrow n+1 = 2^k \Rightarrow \log_2(n) = K.$

i	j	K
1	log(n)	log(n) * log(n)
:	log(n)	log(n) * log(n)
2	log(n)	:
:	:	:
n	log(n)	log(n) * log(n)

T.C $\Rightarrow O(n * \log n * \log n)$

$\Rightarrow O(n \log^2(n)) \rightarrow$ Ans.

(viii) for ($i=1$ to n)

we get $j=n$ times every turn $\therefore i*j = n^2$

k^{th}, Now, $T(n) = n^2 + T(n-3)^2 + (n-6)^2 + \_\_\_ + 1$

Let $k^n - 3k = 1$

$k = (n-1)/3$ tot. terms $= k+1$

$T(n) = n^2 + (n-3)^2 + (n-6)^2 + \_\_\_ + 1$

~~Let $k^n - 3k = 1$~~ $T(n) \approx kn^2$

~~$k = (n-1)/3$~~ $T(n) \approx (n-1)/3 \cdot n^2$

so $T(n) = O(n^3)$

(x) As given n^k & c^n

Relationship be^n n^k & c^n is $n^k = O(c^n)$

$n^k \leq a(c^n)$ $\forall n \geq n_0$ & constant, $a > 0$

for $n_0 = 1$, $c = 2$ $\Rightarrow 1^k < a^2 \Rightarrow n_0 = 1$ & $c = 2$

max level $= n/2^k = 1 \Rightarrow k = \log_2 n$

Date

$$T(n) = C\left(n^2 + (5/16)n^2 + (5/16)^2 n^2 + \ldots + (5/16)^{\log n}\right)$$

$$T(n) = Cn^2 \times 1 \times \left(\frac{1-(5/16)^{\log n}}{1-(5/16)}\right)$$

$$T(n) = Cn^2 + 11/5 \times \left(1-(5/16)^{\log n}\right)$$

$$= O(n^2 C) = O(Cn^2)$$

Q-5) → for

→ for

i	j
1	1
2	$1+3+5$
3	$1+4+7$
|	$1+5+9$
n	

$j = (n-1)/i$ times.

$$\sum_{i=1}^{n} \frac{(n-1)}{i}$$

$$\because T(n) = \frac{(n-1)}{1} + \frac{(n-1)}{2} + \frac{(n-1)}{3} + \cdots + \frac{(n-1)}{n}.$$

$$T(n) = n[1 + 1/2 + 1/3 + \cdots + 1/n] - 1 \times [1 + 1/2 + 1/3 + \cdots + 1/n$$

$\Rightarrow n\log n - \log n$

$T(n) = O(n \log n) \longrightarrow$ Ans.

Q-6)

$\rightarrow$ for

i

2^1

2^k

2^{k^2}

2^{k^3}

$\vdots$

2^{k^m}

where

$2^{k^m} <= n$

$k^m = \log_2 n$

$m = \log_k \log_2 n$

$\therefore \sum_{i=1}^{m} 1$.

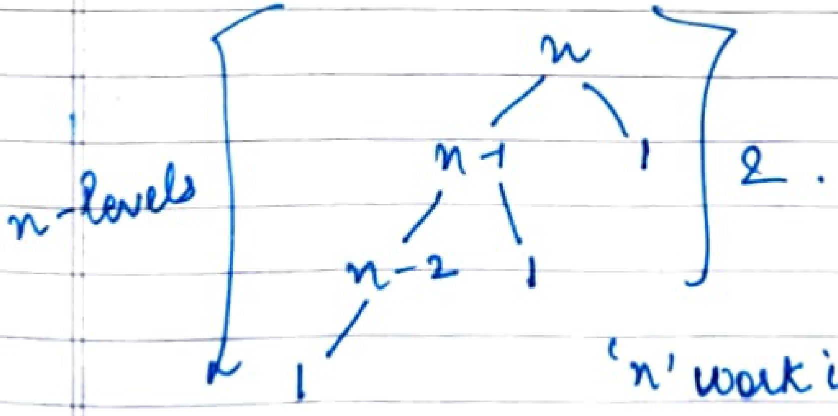
$1 + 1 + 1 - - - m$ times.

$T(n) = O(\log_k \log n) \longrightarrow$ Ans.

Q-7)

$\rightarrow$ Given algorithm divides array in 99% & 1% part

$\therefore T(n) = T(n-1) + O(1)$



'n' work is done at each level..

$T(n) = (T(n-1) + T(n-2) + - - - - + T(1) + O(1)) * n$

$= n \times n$

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

lowest height $= 2$.

highest height $= n$.

$\boxed{\therefore \text{difference} = n-2}$ as $n > 1$

The given algo. produces linear result.