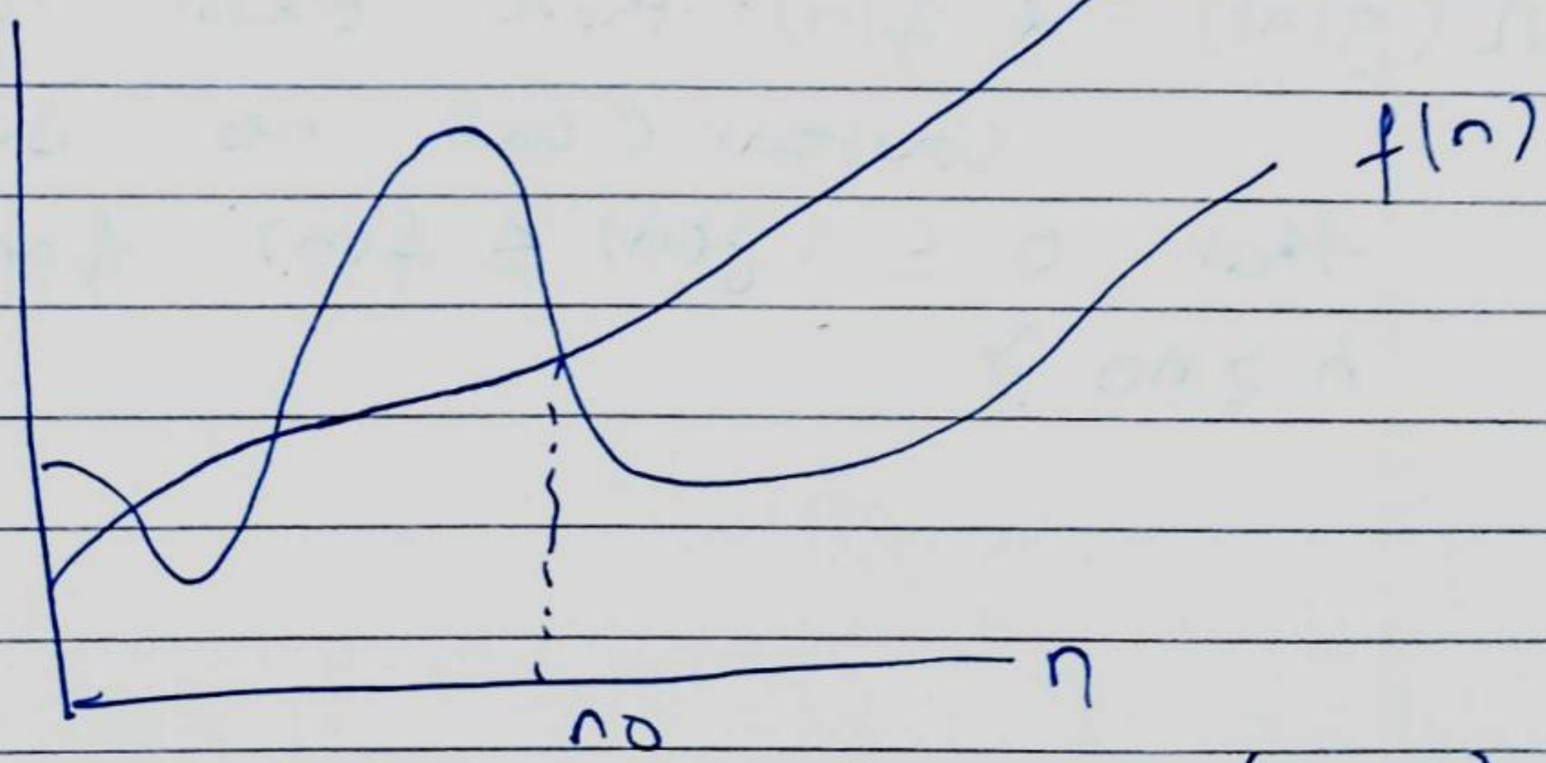
(Tutorial - 1) MUKUL RAWAT
CST- 42
Date: / /

Q1.) Asymptotic Notation :- They are the mathematical notation 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 :-

(i) Big - O - notation.

⊙ provide worst complexity
⊙ provide upper bound of an running time algo.



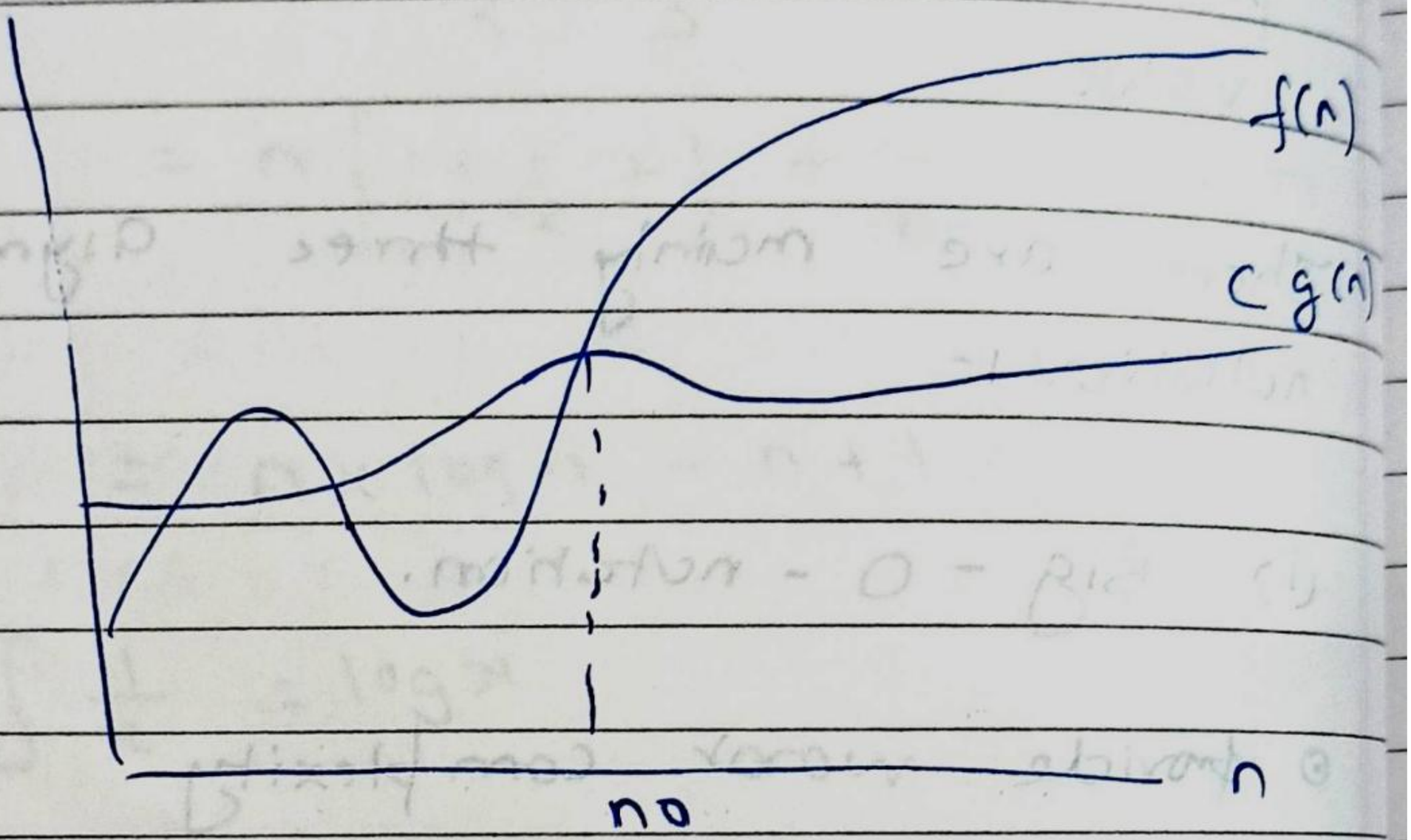
$f(n) = O(g(n))$

$O(g(n)) = \{ f(n) :$ ~~there exist~~ positive constant c & no such that $0 \leq f(n) \leq c g(n)$ for all $n \geq n_0$

Date: 1

(ii) omega Notation (Ω)

•) provide best case complexity of
•) ref. lower bound
running time algo.



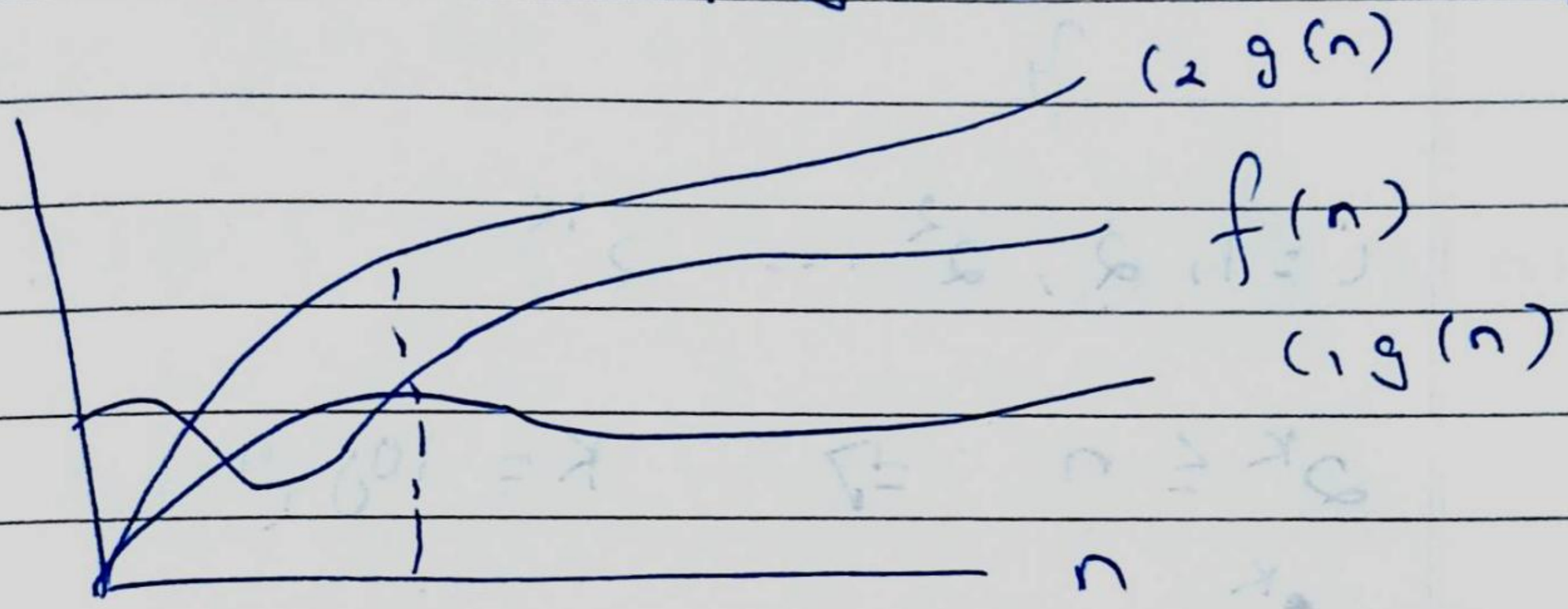
$f(n) = \Omega(g(n))$

$\Omega(g(n)) = \{ f(n)$: there exist positive constant (and no such that $0 \leq (g(n) \leq f(n)$ for all $n \geq no \}$

Date: / /

(iii) theta Notation (Θ- notation)

:) used for analysing avg. time complexity.



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

$\Theta(g(n)) = \{ f(n)$: there exist positive constant c_1, c_2, n_0 such that

$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0 \}$

Mukul Rawat

Date: /

Q2.) for (i = i to n)

{

i = i * 2;

}

$i = 1, 2, 2^2 \cdots 2^k$

$2^k \leq n \Rightarrow k = \log_2 n$

$\because \sum_{i=1}^{k} 1 \Rightarrow 1+1+1 \cdots k \text{ times}$

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

Q3.) $$T(n) = \begin{cases} 3T(n-1) & , n > 0 \\ 1 & , n \leq 0 \end{cases}$$

Using forward substitution:

$T(0) = 1 \quad — \quad 0(1)$

$T(1) = 3T(0)$ — 3

$T(2) = 3^2 T(0)$ — b1

⋮ ⋮

$T(n) = 3^n \times T(0)$ — n

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

=

Date: / /

Q4 $T(n) = \begin{cases} 2T(n-1) - 1, & n > 0 \\ 1, & n \leq 0 \end{cases}$

using forward subs.

$T(1) = 2T(0) - 1, \quad T(0) = 1$

$= 1$

$T(2) = 2 \times T(1) - 1 = 1$

⋮

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

$\therefore T(n) = O(1)$ AN

Mukul Kumar

Mani Kawal

Date: / /

Q5.) int i=1, s=1;

while (s <= n)

{

i++; s = s+i;

printf ('#');

}

for (i=1)	for (i=2)
s = 1+2;	s = 1+2+3

for (i=k)

$1+2+\dots+k <= n$

$\frac{k(k+1)}{2} <= n$

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

$\Rightarrow O(k^2) <= n \Rightarrow k = O(\sqrt{n})$

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

Date: 1

Q6.)

```
void function (int n) {
    int i, count = 0;

    for (int i = 1; i * i <= n; i++)
        count ++ ⟶ O(1)
    }
```

let 'K' be max time value such that

$K^2 \leq n$

$\therefore \quad K \leq \sqrt{n}$

$i^2 \leq n$

$\therefore \quad i \leq 1 \Rightarrow 1+1+ \cdots K \text{ times}$

$i = 1$

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

Mail Rawat

Mawi Rawat

Date: / /

Q7 =

```
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 ++;
}
}
```

let 'm' be highest value of 2^m such that

$2^m <= n \quad \therefore \quad m = \log_2 n$

$\Rightarrow$ for $i = n/2 \quad j = \log n \quad k = \log n$

$i = (n/2+1)$ " "

⋮

$i = n$ " "

$$\therefore \sum_{i=n/2}^{n} j \times k$$

$\Rightarrow n/2 (\log n)^2 \quad \therefore$

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

Mm Rawat

Date: / /

Q8.)

```
func (int n)
{
    if (n==1) return;

    for (i=1 to n)
    {
        for (j=1 to n) {
        printf (...)  →  O(1)
        }
    }
func (n-3);
}
```

for :- for (i=1 to n)

we get j = n times every turn

∴ i x j = n^2

Now, $T(n) = n^2 + T(n-3)$;

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

$T(n-6) = (n-6)^2 + T(n-9)$; $(n-6)^2$ } K terms

⋮

$T(1) = 1$;

Now subs. each value in $T(n)$

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

Date: /

Let

$(n - 3k) = 1$

$\therefore \quad k = (n-1)/3$

$\therefore$ total terms $= k+1$

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

$T(n) \approx n^2 + n^2 + n^2 \ldots$ (k times +1)

$T(n) \approx kn^2$

$T(n) \approx \frac{(n-1)}{3} \times n^2$

$\therefore T(n) = O(n^3)$

Mukul Rawat

Date: / /

Q9.)
```
function (int n)
{
    for (i=1 to n)
    {
        for (j=1; j<=n; j=j+i)
        printf ("*");
    }
}
```

for :- i = 1 $\quad$ j = 1+2 ... (n ≥ j+i)

i = 2 $\quad$ j = 1+3+5 ... "

i = 3 $\quad$ j = 1+4+7 ... "

⋮ $\quad$ ⋮

i^{th} term of ap is

$T(m) = a + d \times m$

$T(m) = 1 + d \times m$

$(n-1)/d = m$

∴ for $\quad$ i = 1 $\quad$ $(n-1)/1$ times (j)

i = 2 $\quad$ $(n-1)/2$ times

i = 3 $\quad$ $(n-1)/3$ times

i = n-1 $\quad$ 1

Date:

We get

$$T(n) = i_1 j_1 + i_2 j_2 \cdots i_{n-1} j_{n-1}$$

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

$$= n + n/2 + n/3 \cdots + \frac{n}{n-1} - n \times 1 +$$

$$= n\left[1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{n-1}\right] - n + 1$$

$$= n \times \log n - n + 1$$

Since $\int \frac{1}{x} = \log x$

$\therefore$ $T(n) = O(n \log n)$

Mst Rawat

Mkm Rawat

Date: / /

Q10.) We have given

n^k & c^n

ay $K \geq 1$ & $c > 1$.

$\Rightarrow$ For values $K \geq 1$, $c > 1$

we have $c^n \geq n^k$

$\therefore$ $n^k = O(c^n)$

$\forall$ $n \geq n_0$, & some constant $K_0 > 0$

.

$\Rightarrow$ $K_0 c^n \geq n^k$

for $c > 1$ & $n = 1$

we get.

$\Rightarrow$ $K_0 c \geq 1$

$\therefore$ $\boxed{c > 1 \quad \& \quad n_0 = 1}$ Ans