Name :- Saumya Negi
Section :- I
Roll No :- 38

TCS-409

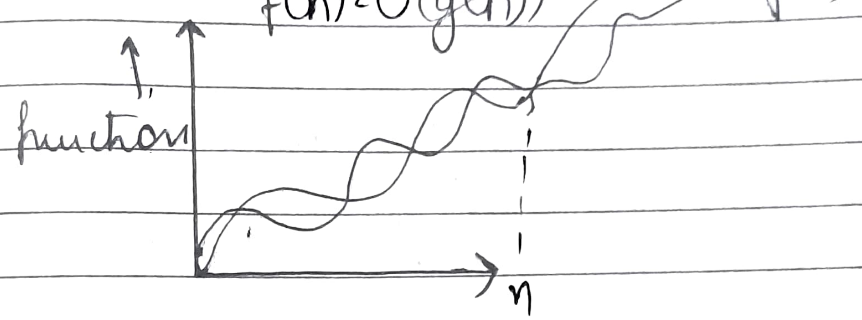
Tutorial 1

Q1 Asymptotic notations means tending to infinity. They are used to tell the complexity when input is very large.

Diff. types of asymptotic notations

1) **Big Oh (O)**

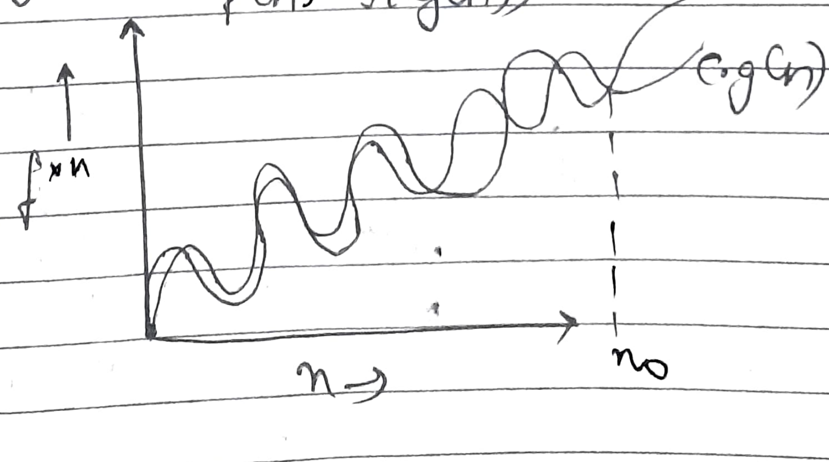
$f(n) = O(g(n))$



Ex
```
for (i=1; i<=n; i++)
{ print ("*"); — O(1)
}
```
$T(n) = O(n)$

2) **Big Omega (Ω)**

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



Ex $f(n) = 2n^2 + 3n + 5$, $g(n) = n^2$

$0 \le c \cdot g(n) \le f(n)$

$0 \le c \cdot n^2 \le 2n^2 + 3n + 5$

$c \le 2 + \frac{3}{n} + \frac{5}{n^2}$

On putting $n = \infty$, $\frac{3}{n} \to \infty$, $\frac{5}{n^2} \to \infty$

$\Rightarrow c = 2$

$\Rightarrow 2n^2 \le 2n^2 + 3n + 5$

On putting $n = 1$

$2 \le 2 + 3 + 5$

$2 \le 10$ True

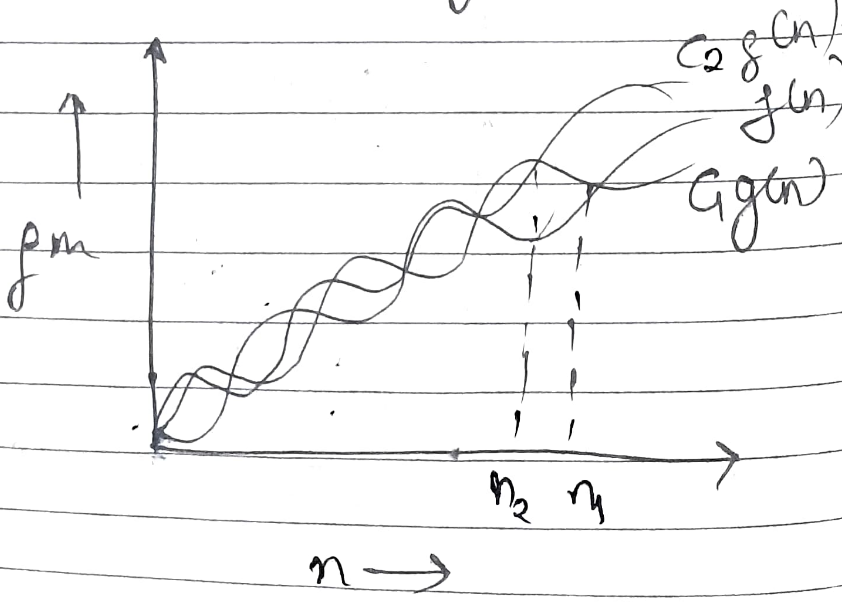
$\Rightarrow \boxed{c = 2,\ n = n_0 = 1}$

$0 \le 2n^2 \le 2n^2 + 3n + 5$

$f(n) = \Omega(n^2)$.

3) Big Theta (Θ)

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



classmate

Date

Page

Ex $f(n) = 10\log_2 n + 4$, $g(n) = \log_2 n$

$f(n) \le c_2 \cdot g(n)$

$\Rightarrow 10\log_2 n + 4 \le 10\log_2 n + \log_2 n$

$10\log_2 n + 4 \le 11\log_2 n$

$c_2 = 11$

$\Rightarrow 4 \le 11\log_2 n - 10\log_2 n$

$4 \le \log_2 n$

$16 \le n$

$\therefore$ $n \ge 16$

$n = 16$

$c_2 = 11$

$f(n) \ge c_1 \cdot g(n)$

$10\log_2 n + 4 \ge 4\log_2 n$

$c_1 = 4, n > 0$

$\Rightarrow n_1 = 1$ $\Rightarrow n_2 = 16$

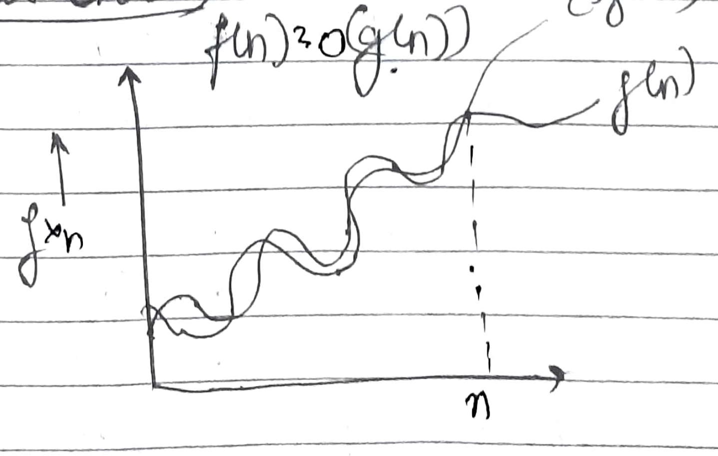
$\Rightarrow \log_2 n \le 10\log_2 n + 4 \le 11\log_2 n$

$c_1 = 4$, $c_2 = 11$

$\Rightarrow \Theta(\log_2 n)$

4) Small Oh (o) -

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



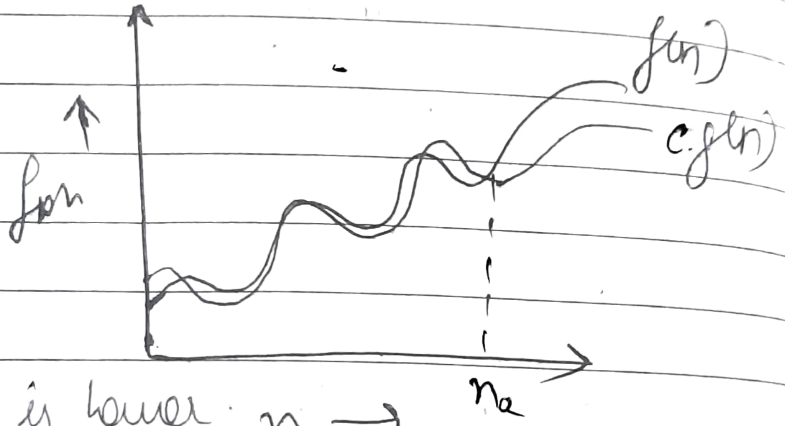
$f(n) = o(g(n))$

iff, $f(n) < c \cdot g(n)$

$\forall\ n > n_1$ and $\forall$ constant, $c > 0$

5) <u>Small omega (ω)</u> –

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



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

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

when,

$f(n) > c \cdot g(n)$

$\forall\ n \cdot n_0$

$\&\ \forall\ c > 0$

Q2 values of $i = \underbrace{1, 2, 4, 6, 16 \ldots\ldots n}_{k \text{ terms}}$

So this is a GP with $a = 1$, $r = 2$

Now,

k^{th} term :– $t_k = ar^{k-1}$

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

$n = 2^{k-1}$

taking $\log_2$ on both sides

$\Rightarrow \log_2 n = \log_2 2^{K-1}$

$\log_2 n = (K-1)\log_2 2$

$\log_2 n = K-1 \Rightarrow K = 1 + \log_2 n$ $[\because \log 2 = 1]$

$\therefore$ Time complexity $T(n) = O(K)$

$= O(1 + \log_2 n)$

$= O(\log_2 n)$

Q3 $T(n) = 3T(n-1)$ --- ①

put $n = n-1$ in eqn ①

$T(n-1) = 3T(n-1-1)$

$T(n-1) = 3T(n-2)$ —— ②

Put value of $T(n-1)$ from eqn ② in eqn ①

$T(n) = 3[3T(n-2)]$

$T(n) = 9T(n-2)$ --- ③

Put $n = n-2$ in eqn ②

$T(n-2) = 3T(n-3)$ --- ④

Put value of $T(n-2)$ in eqn ③

$T(n) = 3[9T(n-3)]$

$T(n) = 27T(n-3)$ —— ⑤

On generalising eqn 5

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

Put $n-K = 0$

$\Rightarrow T(n) = 3^K T(0)$

$= 3^K$ $(\because T(0) = 1)$

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

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

Sol^n.

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

Put $n = n-1$ in eq^n (1)

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

Put value of $T(n-1)$ from (2) in (1)

$$T(n) = 2[2T(n-2) - 1] - 1$$
$$T(n) = 4T(n-2) - 2 - 1 \quad \text{--- (3)}$$

Put $n = n-2$ in eq^n (1)

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

Put value of $T(n-2)$ in eq^n (3)

$$T(n) = 4[2T(n-3) - 1] - 2 - 1$$
$$T(n) = 8T(n-3) - 4 - 2 - 1$$

On Generalising

$$T(n) = 2^k T(n-k) - 2^{k-1} - 2^{k-2} - \cdots - 1$$

Put $n-k = 0 \Rightarrow n = k$, $T(0) = 1$ (Given)

$$T(n) = 2^n T(0) - 2^{n-1} - 2^{n-2} - \cdots - 1$$
$$= 2^n - \underbrace{[2^{n-1} + 2^{n-2} + \cdots + 1]}_{k \text{ terms}}$$

$$\Rightarrow \quad a = 2^{n-1}, \quad r = 1/2$$

Sum of GP = $\dfrac{2^{n-1}\left[1-\left(\frac{1}{2}\right)^{n-1}\right]}{1-1/2}$

$= 2^n - 2$

$\Rightarrow T(n) = \cancel{2^n} - [\cancel{2^n} - 2] = 2$

$= O(2)$

$\boxed{T(n) = O(1)}$

Que.5 What should be the time complexity of-

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

Sol^n:

i	s
1	1
2	3
3	6
⋮	10
⋮	15
⋮	⋮
⋮	n
n	K times

$s = \underbrace{1, 3, 6, 10, 15, \ldots n}_{K \text{ terms}}$

K^{th} term, $t_K = t_{K-1} + K$

$K = t_K - t_{K-1}$ --- (1)

$\Rightarrow K = n - t_{K-1}$

loop runs k-times

Time complexity $= O(1+1+1+n-t_{n-1})$

but $t_{n-1} = c$ (constant)

$\therefore$ Time complexity $= O(3+n-c)$
$= O(n)$

Que-7: Time Complexity of

```
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++
}.
```

Sol^n

$i \to n/2, \frac{n+2}{2}, \frac{n+4}{2}, \frac{n+6}{2}, \text{------ upto } n$

$= \frac{n+0\times2}{2} + \frac{n+1\times2}{2} + \frac{n+2\times2}{2}, \text{---- upto } n$

General form.

$$t_K = \frac{n+K*2}{2}$$

total terms $= K+1$

$$t_{K+1} = n$$

$$\Rightarrow \frac{n+(K+1)*2}{2} = n$$

$$n+2K+2 = 2n$$

$$2K = n-2$$

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

i	j	k
$n/2$	$\log_2 n$ times	$(\log_2 n)^2$
$\frac{n+2}{2}$	$\log_2 n$ times	$(\log_2 n)^2$
⋮	⋮	
n	$\log_2 n$ times	$(\log_2 n)^2$
$(\frac{n}{2}-1)$ times		

$$\Rightarrow \left(\frac{n}{2}-1\right)(\log_2 n)^2$$

$$= O\left(\frac{n}{2}\log_2^2 n - \log_2 n\right)$$

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

Que-6 Time complexity of -

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

sol^n.

$i*i$
1^2
2^2
3^2
4^2
⋮
n

$$i*i = \underbrace{1^2, 2^2, 3^2, 4^2, \dots n}_{K \text{ terms}}$$

$$\Rightarrow K\text{th term} = t_K = K^2$$

$$K^2 = n$$

$$K = n^{1/2}$$

Time complexity $= O(1+1+1+n^{1/2})$
$= O(n^{1/2})$
$= O(\sqrt{n})$

Ques-8: Time complexity of

```
function(int n) {
    If (n==1) return;   — O(1)
    for (i=1 to n) {   — O(n)
        for(j=1 to n) {  — O(n)
            printf("*");  — O(1)
        }
    }
    function(n-3);
}.
```

Sol^n. for function call

$$\underbrace{n, n-3, n-6, n-9 \cdots\cdots 1}_{K \text{ terms}}$$

AP with $d=-3$, $a=n$

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

$1 = n+(K-1)(-3)$

$\frac{1-n}{(-3)} = K-1$

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

$K = \frac{n-1+3}{3}$

$$\boxed{K = \frac{n+2}{3}}$$

Hence ! function have a recursive call $\frac{n+2}{3}$ times

$\Rightarrow$ Time Complexity $= \left(\frac{n+2}{3}\right)(n)(n)$

$= O(n^3)$

Ques-9 Time Complexity of

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

Sol^n

for $i=1 \rightarrow j=1,2,3,4, \cdots\cdots n = n$

for $i=2 \rightarrow j=1,3,5,7, - - - n = n/2$

for $i=3 \rightarrow j=1,4,7, - - - - - n = n/3$

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

$$\Rightarrow \sum_{j=n}^{1} n + n/2 + n/3 + n/4 + - - - + 1$$

$$\sum_{j=n}^{1} n\left[1 + 1/2 + 1/3 + - - - - - - + 1/n\right]$$

$$\sum_{j=n}^{1} n\log n$$

$T(n) = [n\log n]$

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

Sol^n:

As given n^K and c^n

relation b/w n^K and c^n is $\boxed{n^K = O(c^n)}$

as $n^K \leq a c^n$ $\forall\ n \geq n_0$ for a constant $a > 0$

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^K \leq a 2^1$

$\therefore \boxed{n_0 = 1, \text{ and } c = 2}$