Name: Gaurav Sengar
Section: D
Roll No: 03

Tutorial 1

Date.....

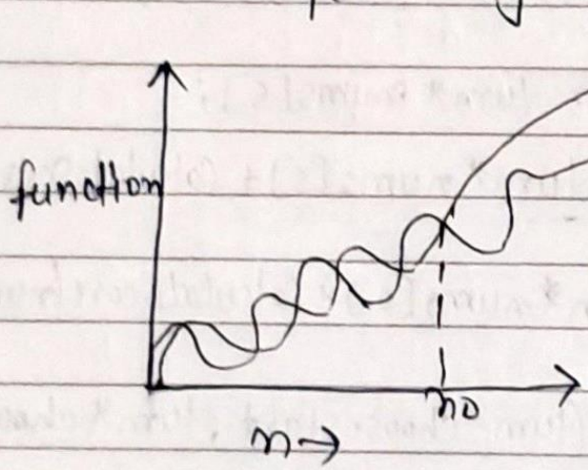
Ques-1: What do you mean by Asymptotic notations. Define different type of notations along with example.

Ans: Asymptotic Notations: Means tending to infinity. They are used to used to tell the complexity when input is very large.

→ Different types of Asymptotic Notations:

1. Big oh (O) Notation:

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



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

$f(n) = O(g(n))$
iff $f(n) \leq c.g(n)$
$\forall\ n \geq n_0$ and some constant,

Example:

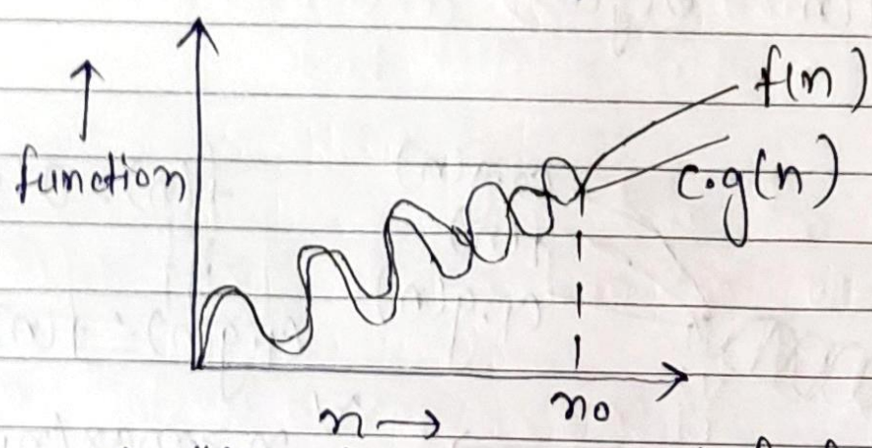
```
for(i=1; i<=n; i++)
    printf("*");  ---- O(1)
```

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

Date.....

2. Big Omega (Ω):

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



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

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

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

$\forall\ n \geq n_0$, and some constant $c > 0$

Example:

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

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

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

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

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

$$\Rightarrow c = 2$$

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

On putting $n = 1$

$$2 \leq 2 + 3 + 5$$

$$2 \leq 10 \quad \text{true.}$$

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

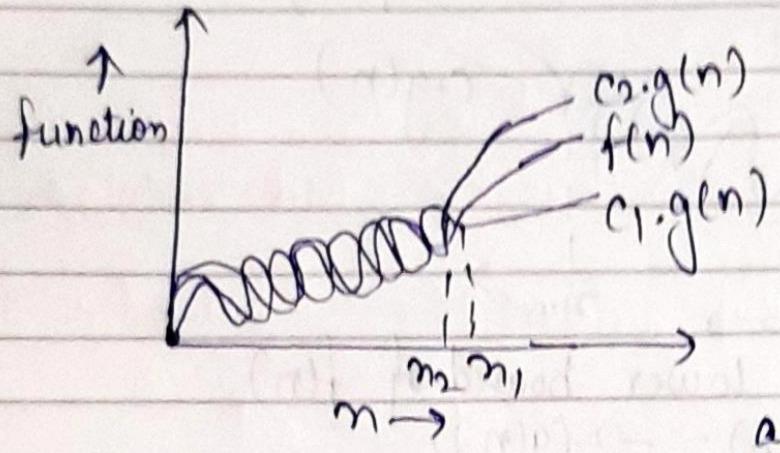
$$0 \leq 2n^2 \leq 2n^2 + 3n + 5$$

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

Date.....

3. Big Theta (Θ):

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



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

iff

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

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

and some constant

$c_1 > 0$ and $c_2 > 0$

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

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

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

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

$$c_2 = 11$$

$$\rightarrow 4 \leq 11\log_2 n - 10\log_2 n$$

$$4 \leq \log_2 n$$

$$16 \leq n$$

Here

$\forall$ $n \geq 16$

$n_2 = 16$

& $c_2 = 11$

Teacher's Sign .....

Date

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

$\Rightarrow n \log_2 n + 4 \geq 1 \log_2 n$

$c_1 = 1, \; n > 0$

$\Rightarrow n_1 = 1 \Rightarrow n_2 = \max(n_1, n_2) \Rightarrow n_2 = 16$

$\Rightarrow \log_2 n \leq n \log_2 n + 4 \leq 11 \log_2 n$

$c_1 = 1, \; c_2 = 11$

$\Rightarrow \Theta(\log_2 n)$

4. Small oh (o):

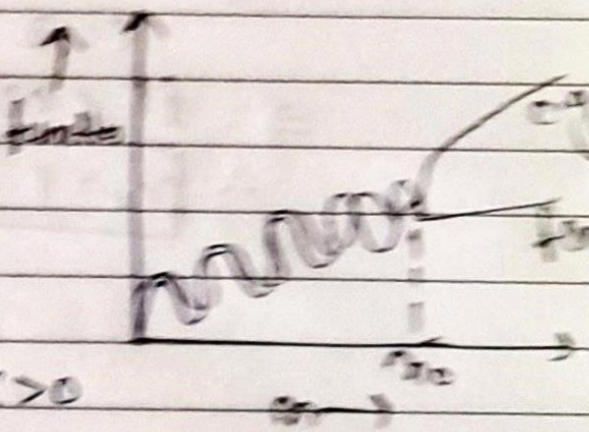
$f(n) = o(g(n))$

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

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

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

$\forall n > n_0$ and $\forall$ constant $c > 0$



5. Small omega (ω):

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

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

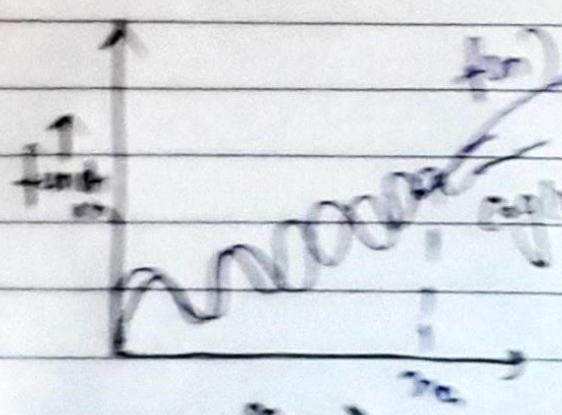
$f(n) = \omega(g(n))$

when

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

$\forall n > n_0$

and $\forall c > 0$



Date

Ques-2: What should be the time complexity of:
for (i=1 ton) { i=i*2; }

Sol^n

values of i = 1, 2, 4, 8, 16, ... n
(K terms)

As this is a G.P 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 to the both sides.

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

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

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

$\therefore$ Time complexity $T(n) = O(K)$
$= O(1 + \log_2 n)$
$= O(\log_2 n)$.

Ques-3:

$T(n) = \{3T(n-1)$ if $n>0$, otherwise $1\}$

Sol^n.

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

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

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

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

Date

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

$$T(n) = 3[3T(n-2)]$$
$$T(n) = 9T(n-2) \quad \cdots\cdots (3)$$

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

$$T(n-2) = 3T(n-3) \quad \cdots\cdots (4)$$

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

$$T(n) = 3[9T(n-3)]$$
$$T(n) = 27T(n-3) \quad \cdots\cdots (5)$$

On Generalising eq^n (5)

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

Put $n-k = 0$

$$\Rightarrow T(n) = 3^k T(0)$$
$$= 3^k \qquad (\because T(0) = 1)$$

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

Date

Ques 4. $T(n) = \begin{cases} 2T(n-1)-1 & \text{if } n>0, \text{ otherwise } 1 \end{cases}$

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)

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

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

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

Date

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

$= 2^n - 2$

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

$= O(2)$

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

Ques-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
1	10
⋮	15
⋮	⋮
1	n
n	K times

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

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

$K = t_K - t_{K-1} \quad \ldots (1)$

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

Date.....

loop runs k-times

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

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

∴ Time complexity = $O(3+n-C)$

$= O(n)$

Ques-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

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

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

General form.

$$t_k = \frac{n + k*2}{2}$$

total terms = k+1

$t_{k+1} = n$

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

$n + 2k + 2 = 2n$

$2k = n - 2$

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

Date

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

$(\frac{n}{2} - 1)$ times

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

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

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

Qw-6 Time complexity of-

```
void function(int n) {
int i, count = 0;          O(1)
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

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

$\Rightarrow$ kth term; $t_k = k^2$

$k^2 = n$

$k = n^{1/2}$

Date

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

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

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

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

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

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

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

Teacher's Sign .....

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

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

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

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

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

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

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

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

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

Date

Qun-10 for the functions, n^k and c^n, what is the asymptotic notation relationship between these functions. Assume that $k>=1$ and $c>1$ are constants. find out the value of c and n_0 for which relation holds.

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 $n^k \leq ac^n$ $\forall\ n \geq n_0$ for a constant $a>0$

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^k \leq a2^1$

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