Name – Shruti Kanchi
Section – AI & DS
Roll No – 2017970

## Tutorial No-1

**Ques 1.** What do you understand by Asymtotic notation define different asymptotic notation with example.

**Ans.**

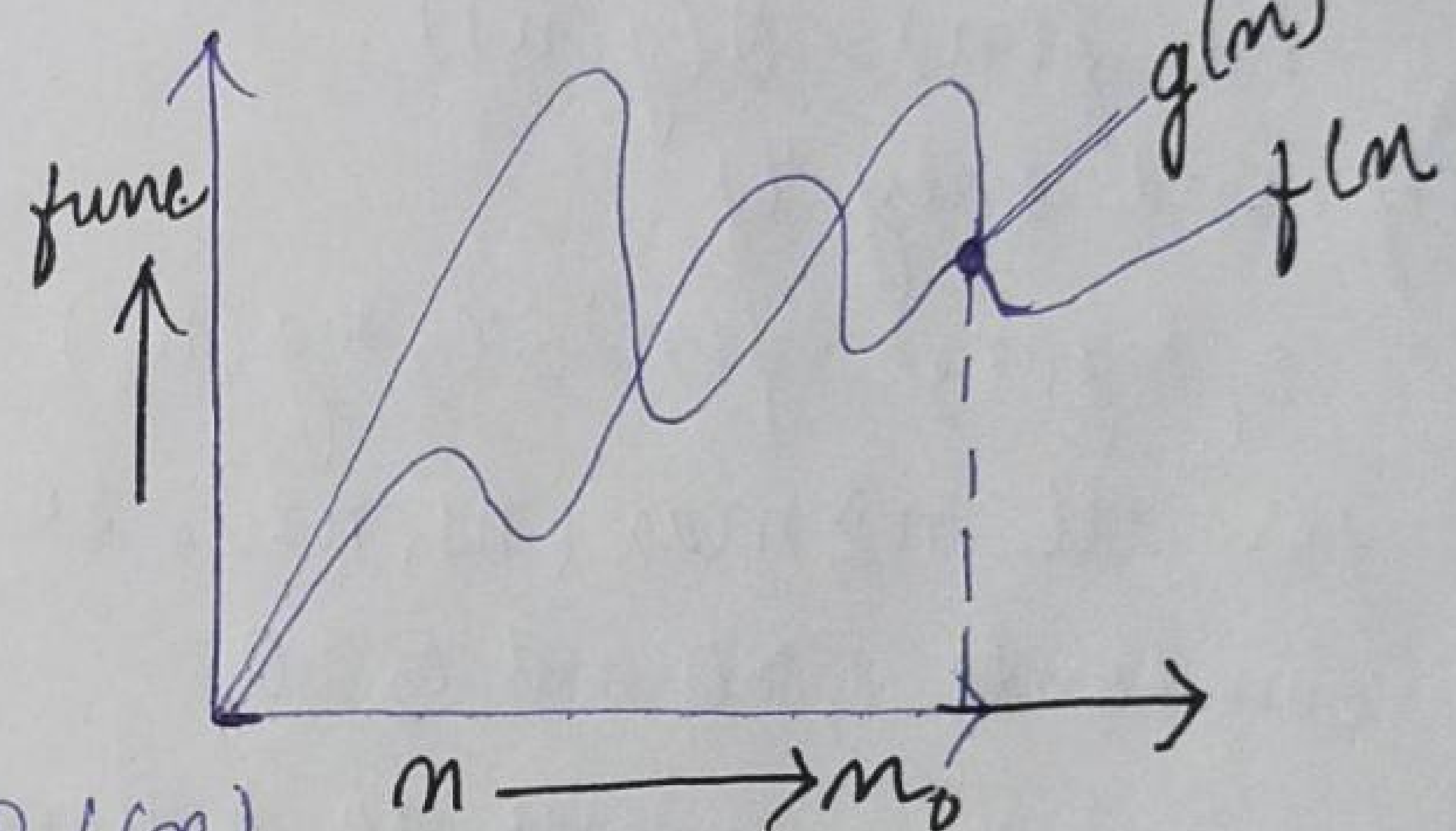
(i) **Big O(n)**

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

if $f(n) \leq g(n) \times c \; \forall \, n \geq n_0$

for some constant, $c > 0$

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




eg:- $f(n) \Rightarrow n^2 + n$

$g(n) = n^3$

$n^2 + n \leq c * n^3$

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

(ii) **Big Omega ($\Omega$)**

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

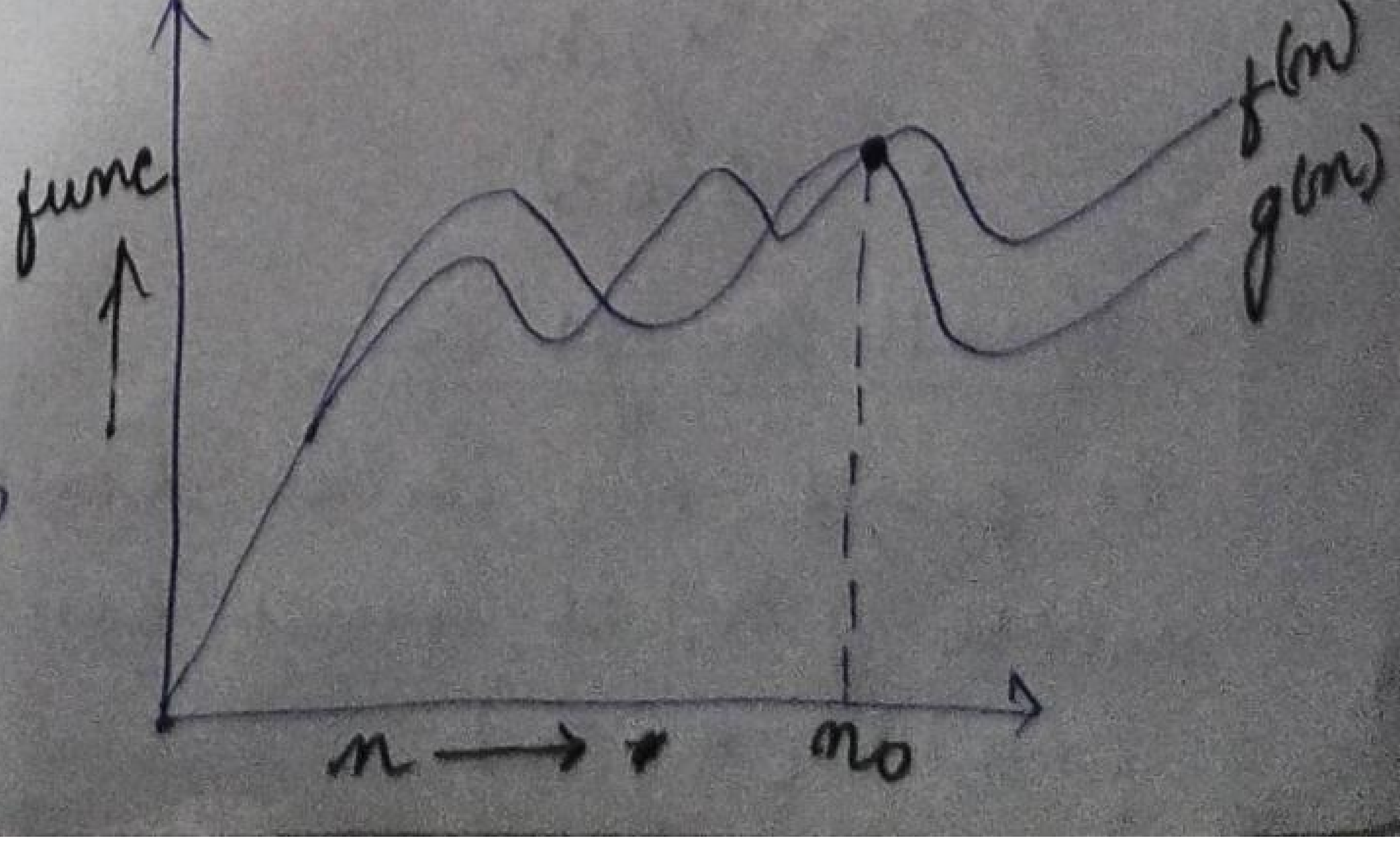
means $g(n)$ is "tight" lowerbound of $f(n)$ i.e $f(n)$ can go beyond $g(n)$

i.e $f(n) = \Omega \, g(n)$

if and only if

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

$\forall \, n_2 > n_0$ and $c = \text{constant} > 0$



Ex - $f(n) \Rightarrow n^3+4n^2$

$g(n) = n^2$

i.e $f(n) \geq c * g(n)$

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

(iii) Big Theta ($\theta$) - when $f(n) = \theta(g(n))$ gives the tight upperbound and lowerbound both

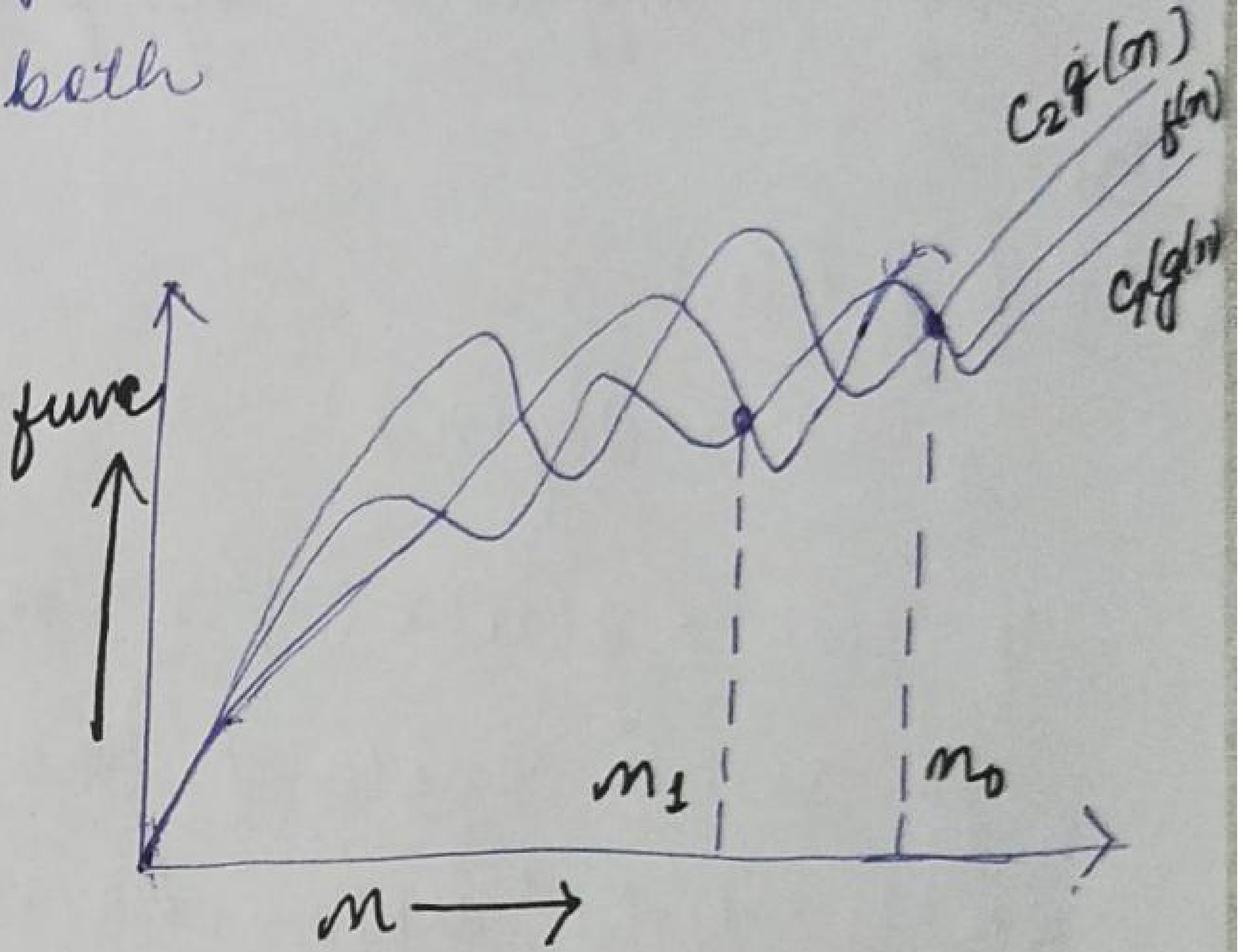
i.e $f(n) = \theta(g(n))$

if and only if

$c_1 * g(n_1) \leq f(n) \leq c_2 * g(n_2)$

for all $n \geq \max(n1, n2)$, some constants $c_1 > 0$ and $c_2 > 0$

i.e $f(n)$ can never go beyond $c_2\, g(n)$ and will never come down of $c_1\, g(n)$.




Ex - $3n+2 = \theta(n)$ as $3n+2 \geq 3n$

$3n+2 \leq 4n$ for $n$, $c_1 = 3$, $c_2 = 4$ & $n_0 = 2$.

(iv) Small o(n).

when $f(n) = o(g(n))$ gives the upper bound.

i.e $f(n) = o(g(n))$

if and only if

$f(n) < c * g(n)$

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

(v) Small Omega ($\omega$)

It gives the lower bound ie $f(n) = \omega g(n)$

where $g(n)$ is lower bound of $f(n)$ if and only if

$f(n) > c * g(n)$

Scanned by TapScanner

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

**Ques 2.** What should be the time complexity of:

```
for (int i = 1 to n)
{
    i = i*2;  ⟶ O(1)
}
```

for $i \Rightarrow 1, 2, 4, 6, 8 \ldots\ldots n$ times

i.e series is a GP

So $a = 1$, $r = 2/1$

$k^{th}$ value of GP

$t_k = ar^{k-1}$

$t_k = 1(2)^{k-1}$

$2n = 2^k$

$\log_2(2n) = k \log 2$

$\log_2 2 + \log_2 n = k$

$\log_2 n + 1 = k$ (Neglecting '1')

So, Time Complexity $T(n) \Rightarrow O(\log n)$ – Ans.

**Ques 3.** $T(n) = \{ 3T(n-1)$ if $n > 0$

otherwise 1.

}

i.e. $T(n) \Rightarrow 3T(n-1)$ —— (1)

$T(n) \Rightarrow 1$.

put $n \Rightarrow n-1$ in (1)

$T(n-1) \Rightarrow 3T(n-2)$ —— (2)

put (2) in (1)

$T(n) = 3 \times 3T(n-2)$

$T(n) \Rightarrow 9T(n-2)$ —— (3)

put $n \Rightarrow n-2$ in (1)

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

put in (3)

$T(n) = 27T(n-3) \longrightarrow$ (4)

Generating series

$T(k) = 3^k T(n-k)$ — (5)

for $k^{th}$ terms let $n-k = 1$ (Base case)

$k = n-1$

put in (5)

$T(n) = 3^{n-1} T(1)$

$T(n) = 3^{n-1}$ (neglecting 3')

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

Ques 4. $T(n) = \{ 2T(n-1) - 1$ if $n>0$,

otherwise 1

}

$T(n) = 2T(n-1) - 1$ —— (1)

put $n = n-1$

$T(n-1) = 2T(n-2) - 1$ —— (2)

put in (1)

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

$= 4T(n-2) - 2 - 1$ —— (3)

put $n = n-2$ in (1)

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

put in (3)

$T(n) = 8T(n-3) - 4 - 2 - 1 \longrightarrow$ 4

Generating series -

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

$k^{th}$ term

Let $n-k = 1$

$k = n-1$

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

$= 2^{n-1} - 2^{k-1}\left(\frac{1}{2} + \frac{1}{2^2} + \ldots + \frac{1}{2^{k-1}}\right)$

is series in GP

$a = 1/2, \ r = 1/2$

So,

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

$= 2^{n-1}\left(1 - 1 + (1/2)^{n-1}\right)$

$= \frac{2^{\cancel{n-1}}}{2^{\cancel{n-1}}} = 1.$

$T(n) = O(1)$ Ans

Ques 5. What should be time complexity of

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

$i = 1, 2, 3, 4, 5, 6$

$s = 1 + 3 + 6 + 10 + 15 + \ldots\ldots$

Sum of $s = 1 + 3 + 6 + 10 + \ldots\ldots + n$ — (1)

Also $S = 1 + 3 + 6 + 10 + \ldots\ldots T_{n-1} + T_n$ — (2)

$0 = 1 + 2 + 3 + 4 + \ldots\ldots n - t_n.$

$t_K = 1 + 2 + 3 + 4 + \ldots\ldots + K.$

$T_K = \frac{1}{2} K(K+1)$

for K iterations

$1 + 2 + 3 + \ldots\ldots K <= n$

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

$\frac{K^2 + K}{2} <= n$

$O(K^2) \leq n.$

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

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

<u>Ques 6</u> Time complexity of

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

As $i^2 = n$

$i = \sqrt{n}$

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

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

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

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

$T(n) = O(n)$ — Ans.

<u>Question 7.</u> Time complexity of.

```
void f (int n)
{
   int i, j, k, count=0;
   For (int i = n/2 ; i<=n; ++i)
   For (j =1 ; j<=n; j=j*2)
   For (k=1; k<=n; k=k+2)
     count ++;
}
```

Since, for $k = k^2$

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

∵ series is in GP

So $a=1$, $r=2$

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

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

$n = 2^k - 1$

$n+1 = 2^k$

$\log_2(n) = k$

| i | j | K. |
|---|---|---|
| 1. | log(n) | log(n) * log(n) |
| ⋮ | log(n) | log(n) * log(n) |
| 2. | ⋮ | ⋮ |
| ⋮ | ⋮ | ⋮ |
| n. | log(n) | log(n) * (log n)) |

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

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

Ques 8. Time complexity of

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

for (i=1 to n)

we get j=n times every turn

$\therefore i * j = n^2$

Now,

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

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

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

and $T(1) = 1;$

Now substitute each value in $T(n)$

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

let

$n - 3k = 1$

$\frac{n-1}{3} = k.$ Total turns $= k+1$

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

~~let~~ $T(n) \simeq kn^2$

$T(n) \simeq (k-1)/3 * n^2$

So,

$T(n) = O(n^3)$ — Ans

Ques 9. Time complexity of

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

For $i=1$ $\quad j = 1+2+ \ldots\ldots (n \geq j+i.)$

$i = 2$ $\quad j = 1+3+5 \ldots\ldots (n \geq j+i)$

$i = 3$ $\quad j = 1+4+7 \ldots\ldots (n \geq j+i)$

nth term of AP is

$T(n) = a + d * m.$

$T(m) = 1 + d * m.$

$(n-1)/d = m$

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

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

$\vdots$

$i = n-1$

we get

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

$= \left(\frac{n-1}{1}\right) + \left(\frac{n-2}{2}\right) + \left(\frac{n-3}{3}\right) + \ldots\ldots 1.$

$= n + n/2 + n/3 + \ldots\ldots n/n-1 - n \times 1.$

$= n\left[1 + 1/2 + 1/3 + \ldots\ldots 1/n-1\right] - n*1$

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

Since $\int 1/x = \log x$

$T(n) = O(n \log n)$ —Ans.

Ques 10. For the function $n^{-1}k$ and $c^n$. What is the asymptotic Relationship b/w three functions?
Assume that $K >= 1$ and $C > 1$ are constants. Find out the value of C & no. of which relationship holds.

As given $n^K$ and $c^n$

Relationship b/w $n^K$ and $c^n$ is

$$n^K = O(c^n)$$

$$n^K \leq a(c^n)$$

$$\forall\ n \geq n_0 \ \&\ \text{constant } a > 0$$

for $n_0 = 1,\ c = 2$

$\Rightarrow 1^K < a^2$

$\Rightarrow n_0 = 1 \ \&\ c - 2$