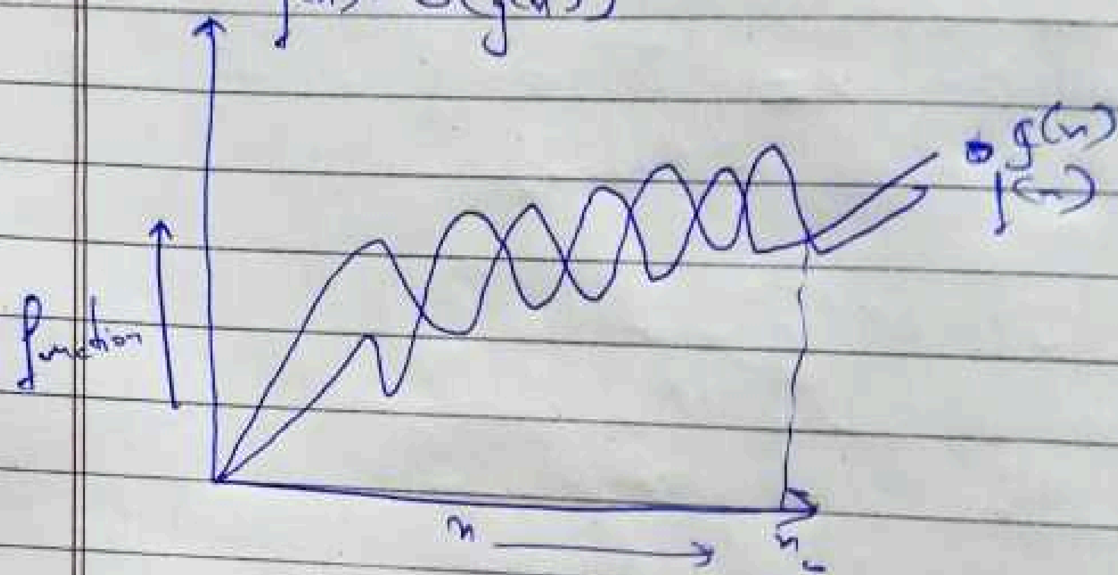
Tutorial-1

Q1.) Ans Asymptotic Notation: ~~They~~ They are the mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value.

Different asymptotic notations:

i) Big O(n)

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



(size of input)

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

if, $f(n) \le c\,g(n)$ $\forall$ $n \ge n_0$

for some constant, $c > 0$

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

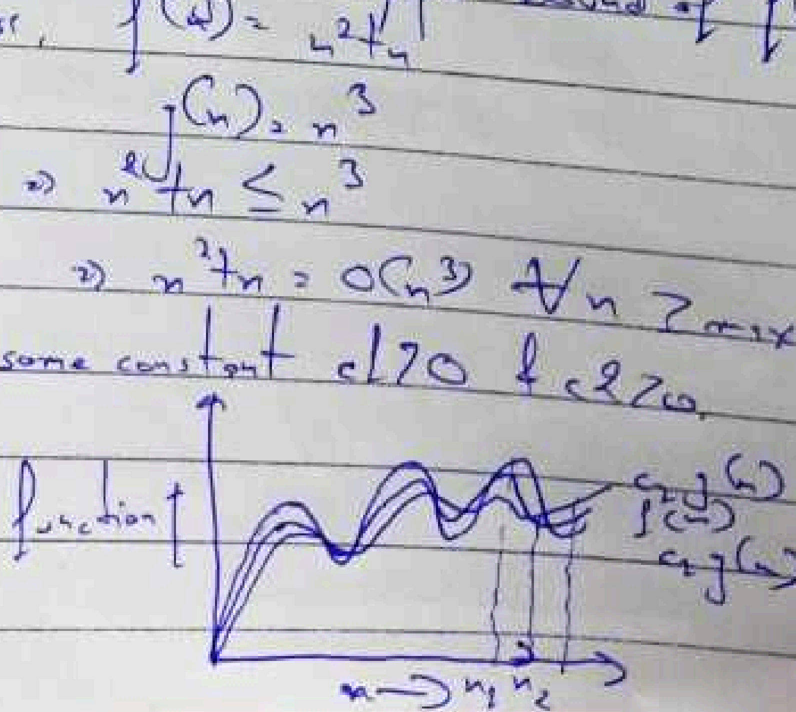
eg, $f(n) = n^2 + n$

$g(n) = n^3$

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

$\Rightarrow n^2 + n = O(n^3)$ $\forall n \ge \max(n_1, n_2)$

for some constant $c_1 > 0$ & $c_2 > 0$.



$\underline{Ex}$:

$3n+2 = \Theta(n)$ as $3n+2 \geq 3n$ &

$3n+2 \leq 4n$ for n, $k_1 = 3$, $k_2 = 4$ & $n_0 = 2$.

iii) Small o (o)

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

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

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

when $f(n) < cg(n)$ $\forall n \geq n_0$ & $\forall$ constants

ex: $f(n) = n^2$

$g(n) = n^3$

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

iv) Big Omega (Ω)

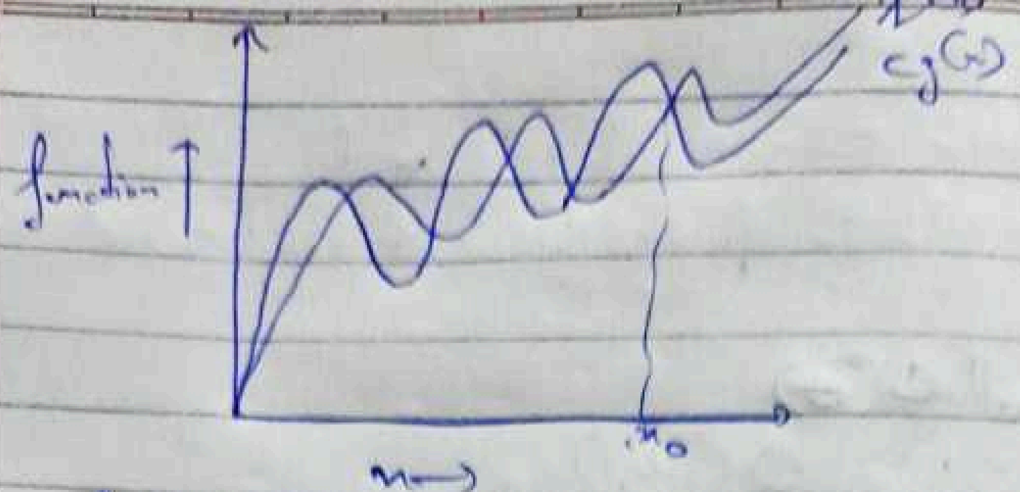
$f(n) = \Omega(g(n))$

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

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

iff $f(n) \geq cg(n)$ $\forall n \geq n_0$

for some constant $c > 0$



$f(n) = n^3 + 4n^2$

$g(n) = n^2$

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

ix) Big theta (Θ)

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

$g(n)$ is both 'tight' upper bound and lower bound of function $f(n)$.

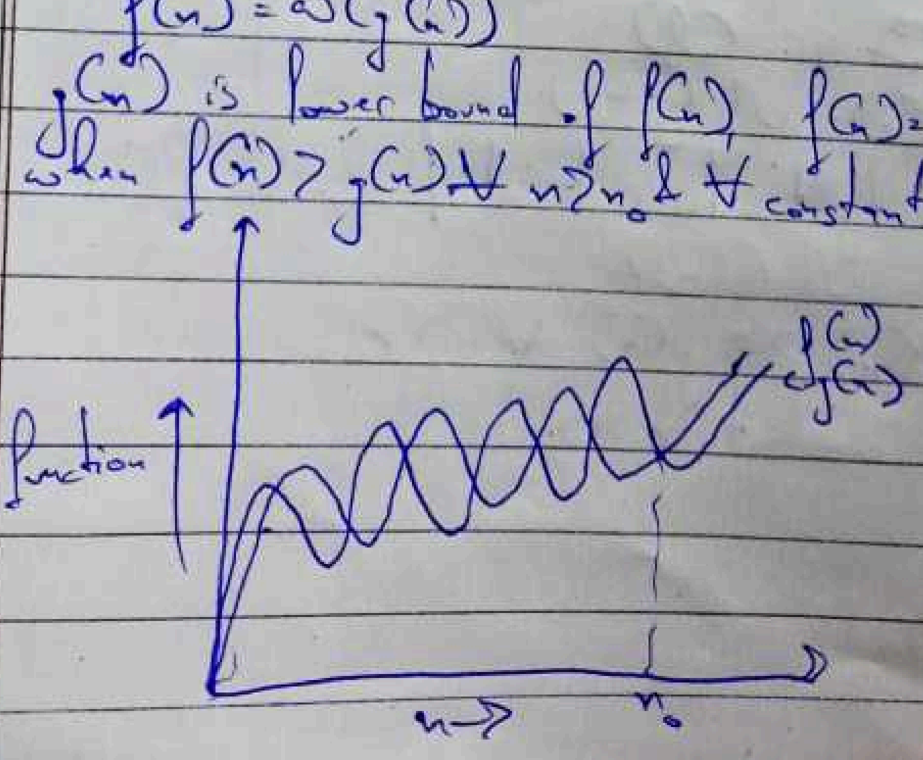
$f(n) = \Theta(g(n))$

iff, $c_1 g(n) \leq f(n) \leq c_2 g(n)$

x) Small omega (ω)

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

$g(n)$ is lower bound of $f(n)$, $f(n) = \omega(g(n))$ when $f(n) > g(n)\ \forall\ n > n_0$ & $\forall$ constants, such that $c > 0$.



$f(n) = 4n+6$

$g(n) = 1$

Q2) A- for (i=1 to n)
{ i = i * 2 }

=> i = $\underbrace{1, 2, 4, 8, 16, \ldots \ldots n}_{k}$ {GP}

$- O(k)$

$a = 1, r = \frac{2}{1} = 2$

GP, k^{th} value $= t_k = a_n = ar^{k-1}$

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

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

$\Rightarrow 2n = 2^k$

$\log 2n = k \log_2 2$

$\Rightarrow k = \log_2 2n$

$\Rightarrow k = \log_2 2 + \log_2 n$

$\Rightarrow k = 1 + \log_2 n$

$\therefore$ Time complexity $= O(1 + \log_k n) = O(\log_2 n)$

Q3) Ans $T(n) = 3T(n-1)$ - ①

Let $n = n-1$

$T(n-1) = 3T(n-2)$ - ②

Put ② in ①

$T(n) = 3 \times 3T(n-2)$ - ③

Put $n = n-2$

$T(n-2) = 3T(n-3)$ - ④

Put ④ in ③

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

$\therefore T(n) = 3^n T(n-n) \Rightarrow T(n) = 3^n T(0) \Rightarrow T(n) = 3^n$

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

PAGE ______
DATE ______

(Q4.) A $T(n) = 2T(n-1) - 1$

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

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

$= \cancel{2^3(}T\ 2^3T(n-3) - 2^2 - 2^1 - 2^0$

$= 2^nT(n-n) - 2^{n-1} - 2^{n-2} - 2^{n-3} - \dots 2^2 - 2^1 - 2^0$

$= 2^n - 2^{n-1} - 2^{n-2} - 2^{n-3} - \dots - 2^2 - 2^1 - 2^0$

$= 2^n - (2^n - 1)$

$T(n) = 1$

(Q5.) A

```
int i=2, s=1;
while (s<=n){
  i++; s=s+i;
  printf("#");
}
```

→ i is incrementing by one step
s is incrementing by value of i
∴ after few iterations
→ k=i=2, s=3 1st iteration
i=3, s=6 2nd iteration
i=4, s=10 3rd iteration
Let the value of n be k.
Values of s = 1, 3, 6, 10, ____
s represents a series of sum of first n natural numbers
for $l=k \Rightarrow s = \frac{k(k+1)}{2}$

∴ for stopping loop, $\frac{k(k+1)}{2} \geq n \Rightarrow k^2+k \geq n$

∴ $\boxed{T(n) = O(\sqrt{n})}$

Q6) A void function(int n){
int i, count = 0;
for (i=1, i*i <= n, i++)
count++;

i = 1, 2, 3, ..., n

$i^2 = 1, 4, 9, \ldots n^2$

So $i^2 <= n$ or $i <= \sqrt{n}$

=> $a_k = a + (k-1)d$

$a = 1$, $d = 1$

=> $a_k <= \sqrt{n}$

$\sqrt{n} = 1 + (k-1).1$

=> $\sqrt{n} = k$

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

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++;
      }
    }
  }
```

$i = \frac{n}{2}$, $j = \log_2 n$, $k = \log_2 n$

$\left(\frac{n}{2}+1\right)$ times $\quad \log_2 n \quad \log_2 n$

$O(i * j * k) = O\left(\left(\frac{n}{2}+1\right) * \log_2 n * \log_2 n\right)$

$O(i * j * k) = O\left(\left(\frac{n}{2}+1\right) * (\log_2 n)^2\right)$

$T(n) = O(n(\log_2 n)^2)$

Q8) Ans

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

$T(n) = T(n-3) + n^2$ — ①

$T(1) = 1$ — ②

Put $n = n-3$ in ①

$T(n-3) = T(n-6) + (n-3)^2$ — ③

Put ③ in ①

$T(n) = T(n-6) + (n-3)^2 + n^2$ — (4)

Put $n = n-6$ in (1)

$T(n-6) = T(n-9) + (n-6)^2$ — (5)

Put (5) in (4)

$T(n) = T(n-9) + (n-6)^2 + (n-3)^2 + n^2$

Generalising,

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

Let $n - 3k = 1$

$\Rightarrow \frac{n-1}{3} = k$

$T(n) = T(1) + \left(n - 3\left(\frac{n-1}{3} - 1\right)\right)^2 + \left(n - 3\left(\frac{n-1}{3}\right)\right)^2 + \dots + n^2$

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

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

$\Rightarrow T(n) = 1 + 4^2 + 7^2 + \dots + n^2$

$\Rightarrow T(n) = n^2 + \dots + 1$

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

Q.) A.

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

for $i=1$, $j = n$ times

for $i=2$, $j = 1+3+5+ \dots + n$

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

$a = 1, d = 2$

$n = 1 + (k-1) * 2$

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

$\boxed{k = \frac{n+1}{2}}$, no. of terms

for $i = 2$, $j = \frac{n+1}{2}$ terms

for $i = 3$, $j = 1 + 4 + 7 + \_\_ + n$

$n = 1 + (k-1) * 3$

$\boxed{\frac{n-1}{3} + 1 = k}$ no. of terms

for $i = 3$, $j = \frac{n+2}{3}$ terms

Generalising

for $i = n$, $j = \frac{n+k-1}{k}$ ~~times~~ times

Times complexity is

$n + \frac{n+1}{2} + \frac{n+2}{3} + \_\_ + \frac{n+k-1}{k}$

$\hookrightarrow$ General terms $= \frac{n+k-1}{k}$

$\therefore \sum_{k=1}^{n} \frac{n+k-1}{k} \Rightarrow$ Gener. Sum $= \frac{\sum_1^n n + \sum_1^n k - \sum_1^n n}{k}$

$\therefore$ Sum $= \frac{n(n+1)/2 + nk - n}{k}$

$$\text{Sum} = \frac{n^2+n/2+nk-n}{k}$$

Neglecting constant terms & terms of lower order

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

Q10.) A. As given n^k & c^n.

Relation between n^k & c^n is

$n^k = O(c^n)$. As $n^k \le a c^n \; \forall n \ge n_0$ & some constant $a>0$

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^k \le a 2^1$

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