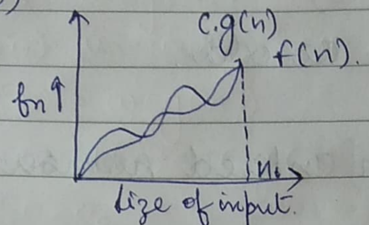
M T W T F S S
Page No.:
Date:
YOUVA

1. Asymptotic Notation:

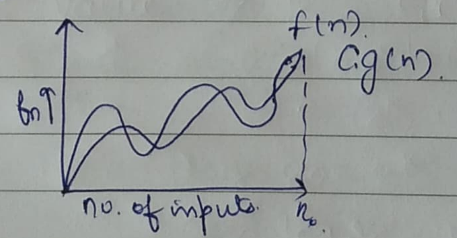
Asymptotic means tending to infinity.
They help us find the complexity of algorithm when input is very large.

1) Big O(O)



$f(n) = O(g(n))$
if $f(n) \le c.g(n)$
$\forall\ n \ge n_0$.
for some constant $c>0$.
$\Rightarrow g(n)$ is tight upper bound of $f(n)$.

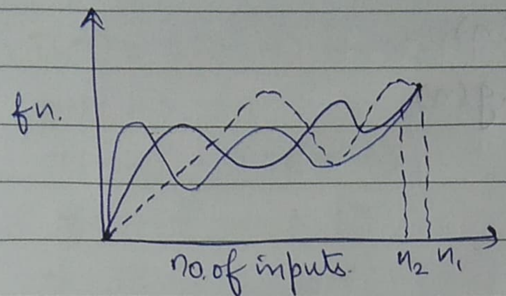
2) Big Omega (Ω).



$f(n) = \Omega(g(n))$
$g(n)$ is tight lower bound of $f(n)$.
$f(n) = \Omega(g(n))$
if $f(n) \ge cg(n)$.
$\forall\ n \ge n_0$ for some constant $c>0$.

Vishal

3) Theta (Θ).



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

g(n) is both tight upper and lower bound of fn f(n).

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

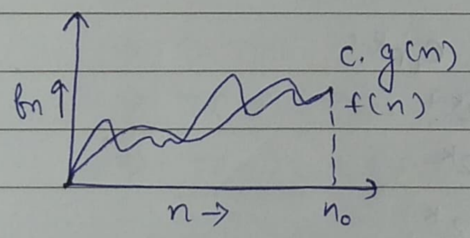
if

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

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

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

4) Small o(o)



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

g(n) is upper bound of fn (f(n)).

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

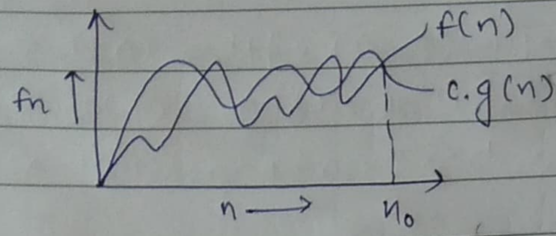
when $f(n) < c.g(n)$

$\forall n > n_0$

and $\forall c > 0$.

vishal

5) Small omega (ω)



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

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

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

when $f(n) > c.g(n)$

$\forall \; n > n_0$

& $\forall \; c > 0$.

Ans Q.

for(i=1 to n) // $i = 1, 2, \cancel{8}, 4, 8, \ldots\ldots n$.

{ i = i*2 } // $O(1)$.

$\Rightarrow \sum_{i=1}^{n} 1+2+4+8+\ldots\ldots n$.

GP k^{th} value $\Rightarrow T_k = c_1 r^{k-1}$

$\Rightarrow 1 \times 2^{k-1}$

$\Rightarrow n = 2^k$

$\Rightarrow 2n = 2^k$

$\Rightarrow \log 2n = k \log 2$

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

$\Rightarrow \log n + 1 = k$.

$\Rightarrow O(k) = O(1 + \log n)$

$= O(\log n)$.

Vishal

M T W T F S S
Page No.:
Date:
YOUVA

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

$T(n) = 3T(n-1)$ — ①

put $n = n-1$

$T(n-1) = 3T(n-2)$ — ②

from ① and ②.

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

$= 9T(n-2)$ — ③.

putting n: n-2 in ①

$T(n) = 3(T(n-3))$ — ④

$T(n) = 27(T(n-3))$

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

putting $n-k=0$.

$\Rightarrow n=k$

$\Rightarrow T(n) = 3^n[T(n-n)]$

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

$\Rightarrow T(n) = 3^n \times 1$

$\Rightarrow \underline{T(n) = O(3^n)}$

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

$T(n) = 2T(n-1) - 1$ — ①

let $n = n-1$

$T(n-1) = 2T(n-2) - 1$ — ②

from ① and ②

$T(n) = 2[2T(n-2) - 1] - 1$.

$T(n) = 4T(n-2) - 3$ — ③.

let $n = n-2$

$T(n-2) = 2T(n-3) - 1$ — ④

vishal

from 3 and 4,

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

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

$\Rightarrow T(n) = 2^K T(n-K) - 2^{K-1} - 2^{K-2} - 1$

$\Rightarrow GP = 2^{K-1} + 2^{K-2} + 2^{K-3} + \dots + 1$

$a = 2^{K-1}$

$r = 1/2$

$\Rightarrow S_K = \dfrac{a(1-r^n)}{1-r}$

$= \dfrac{2^{K-1}(1-(1/2)^n)}{\frac{1}{2}}$

$= \dfrac{2^{K-1}(1-(1/2)^K)}{1/2}$

$= 2^K - 1$

let $n-K=0$

$\Rightarrow n = K$

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

$\Rightarrow T(n) = 2^n \cdot 1 - (2^n - 1)$

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

$\Rightarrow T(n) = O(1)$

Ans 5.

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

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

$s = 1+3+6+10+15+21+\dots n$

Vishal

sum of $s = 1 + 3 + 6 + 10 + \dots\dots + n$ —— ①

also $s = 1 + 3 + 6 + 10 + \dots\dots + T_{n-1} + T_n$ —— ②

from ① and ②

$0 = 1 + 2 + 3 + 4 + \dots\dots + n - T_n$.

$\Rightarrow T_k = 1 + 2 + 3 + 4 + \dots\dots k$

$\Rightarrow T_k = \frac{1}{2} k(k+1)$.

for k iterations

$1 + 2 + 3 + \dots\dots + k <= n$

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

$\Rightarrow \frac{k^2 + k}{2} <= n$

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

$\Rightarrow k = O\sqrt{n}$

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

Ans6. as $i^2 <= n$.

$\Rightarrow i <= \sqrt{n}$.

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

$\sum_{i=1}^{n} 1 + 2 + 3 + 4 + \dots + \sqrt{n}$.

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

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

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

vishal

Ans 7

```
void fn(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++;
}
```

for k = k*2

k = 1, 2, 4, 8, ... n.

$\Rightarrow$ GP $\rightarrow$ a = 1 , r = 2.

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

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

$n = 2k.$

$\log n = k.$

i	j	k
1	log n	log n × log n
2	log n	log n × log n
⋮	⋮	⋮
n	log n	log n × log n

$\Rightarrow O(n * \log n * \log n)$

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

vishal

Ans 8. Function (int n)
{ int (n==1)
return; // O(1)
for (i=1 to n) // i = 1, 2, 3, 4, ... n ⇒ Θ(n).
{ for (j=1 to n) // j = 1, 2, 3, 4, ... n ⇒ Θ(n²)
{ print ('*');
}
}
function (n-3); T(n/3)
}

$\Rightarrow T(n) = T(n/3) + n^2$.

$\Rightarrow a = 1, \quad b = 3. \qquad f(n) = n^2.$

$c = \log_3 1 = 0$.

$\Rightarrow n^0 = 1 > (f(n) = n^2)$.

$\Rightarrow T(n) = \Theta(n^2)$.

Ans 9. void function (int n)
{ for (i=1 to n) // O(n)
{ for (j=1; j<=n; j=j+i)
print ("*") // O(1)
}
}

for $i=1 \Rightarrow j = 1, 2, 3 \ldots\ldots n \quad = n$.
for $i=2 \Rightarrow j = 1, 3, 5 \ldots\ldots n \quad = n/2$
for $i=3 \Rightarrow j = 1, 4, 7, \ldots\ldots n \quad = n/3$.

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

Vishal

$$\Rightarrow \sum_{j=n}^{1} n + \frac{n}{2} + \frac{n}{3} + \frac{n}{4} + \cdots + 1$$

$$\Rightarrow \sum_{j=n}^{1} n\left[1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n}\right]$$

$$\Rightarrow \sum_{j=n} n[\log n]$$

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

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

Ans 10.

as given n^k & c^n

relation b/w n^k and c^n is

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

as $n^k \leq ac^n$

$\forall\, n \geq n_0$ for some constant $a > 0$.

for $n_0 = 1$

$c = 2$

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

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

vishal