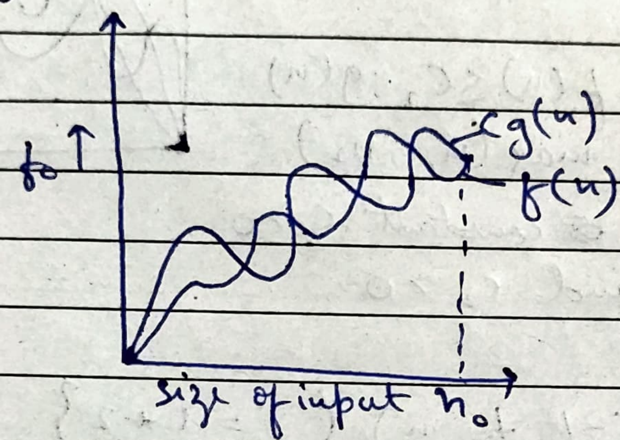
1: Asymptotic Notations - They help us to find the complexity of algorithm when input is very large.

Asymptotic $\rightarrow$ tending to infinity

i) Big O (O)



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

iff $f(n) \leq c \cdot g(n) \quad \forall \; n \geq n_0$

$\Rightarrow$ $g(n)$ is tight upper bound of $f(n)$

ii) Big Omega (Ω)

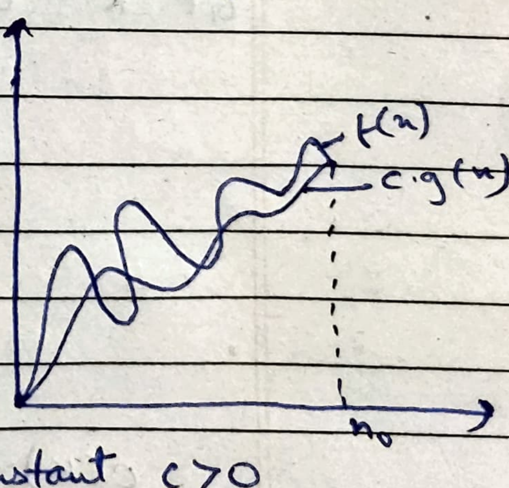
$f(n) = \Omega(g(n))$

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

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

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

$\forall \; n \geq n_0$ for some constant $c > 0$



iii) Theta (θ)

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

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

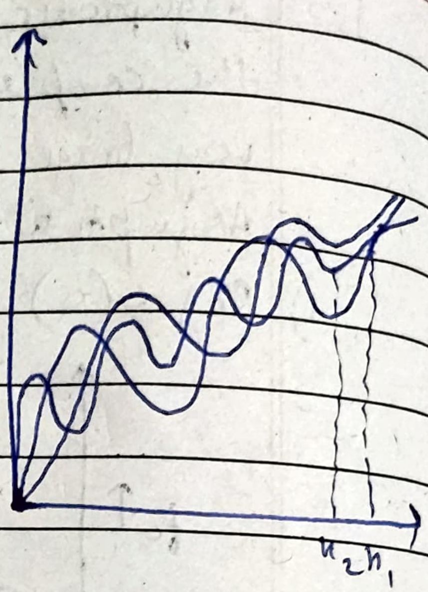
$f(n) = \theta(g(n))$

iff

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

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

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



2. for(i=1 to n) { i = i*2 }

$\sum_{i=1} 1 + 2 + 4 + 4 + \text{---} + n$

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

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

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

$\Rightarrow 2n = 2^k$

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

$\Rightarrow \log_2 1 + \log n = k \log 2$

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

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

$= O(\log n)$

Page No: 1

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

$T(n) = 3T(n-1)$ — 1

Put $n = n-1$ in 1

$T(n-1) = 3T(n-2)$ — 2

Put 2 in 1

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

$= 9T(n-2)$ — 3

Put $n = n-2$ in ~~2~~ 1

$T(n-2) = \cancel{9T(n}\ 3T(n-3)$ — 4

Put 4 in 3

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

$= 27T(n-3)$ — 5

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

Put $n-K = 0$

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

$= 3^n [T(0)]$

$= 3^n$

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

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

$T(n) = 2T(n-1) - 1$ —— (1)

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

$T(n-1) = 2T(n-2) - 1$ —— (2)

Put (2) in (1)

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

$= 4T(n-2) - 2 - 1$ —— (3)

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

Let $k = n$

$T(n) = 2^n T(0) - 2^n + 1$

$= 2^n - 2^n + 1$

$= 1$

$\Rightarrow O(1)$

5.
```
int i=1, s=1;
while (s<=n) {
    i++; s=s+i;
    printf("#");
}
```

i	s	
1	1	=1
2	1+2	=3
3	1+2+3	=6
⋮	⋮	
	n	

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

for k iteration

$1+2+3+ --- +k <= n$

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

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

$O(k^2) <= n$

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

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

6.
```
void function (int n)
{ int i, count =0;
  for (i=1; i*i <= n ; i++)
  { count ++; }
}
```

$i * i <= n$

$i <= \sqrt{n}$

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

$\sum_{i=1}^{n} 1 + 2 + 3 + --- + \sqrt{n}$

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

$O(n)$

7.
```
void function (int n)
{ int i, j, k, count = 0;
  for (i = n/2; i<= n; i++)
  { for (j=1; i<=n; j = j*2)
    { for (k=1; k<=n; k = k*2)
        count ++;
    }
  }
}
```

Page No: 1
Date: / /

for $K = K*2$

$K = 1, 2, 4, 8, \ldots, n$

$n = \frac{a(r^k - 1)}{r-1}$ $(a=1, r=2)$

$n = 2^K - 1$

$\log n = K$

i	j	K
1	$\log n$	$\log n \cdot \log n$
2	$\log n$	$\log n \cdot \log n$
3	$\log n$	$\log n \cdot \log n$
⋮	⋮	⋮
n	$\log n$	$\log n \cdot \log n$

So, $O(n \log^2 n)$

8. int function (int n)

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

Page No:
Date:

$$T(n) = T(n-3) + n^2 \quad \text{--- (1)}$$

Put $n = n-3$

$$T(n-3) = T(n-6) + n^2 \quad \text{--- (2)}$$

$$T(n) = T(n-6) + n^2 + n^2 \quad \text{--- (3)}$$

$\vdots$

$$T(n) = T(n-3K) + Kn^2$$

Put $n - 3K = 1$

$$T(n) = T(1) + Kn^2$$

So, $O(n^2)$

10. For function n^K and c^n what is asymptotic relationship b/w these functions?

Assume that $K >= 1$ and $c > 1$ are constraints. Find out the value of c and n_0 for which relation holds.

as given n^K and c^n
relation b/w n^K and c^n is

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

as $n^K <= ac^n$

$\forall$ $n \geq n_0$ and some constant $a > 0$

for $x_0 = 1$

$c = 2$

$\Rightarrow 1^k \leq a\, 2^1$

So, $x_0 = 1$, $c = 2$