Santhak Parashar
Section – ML
Roll – 51

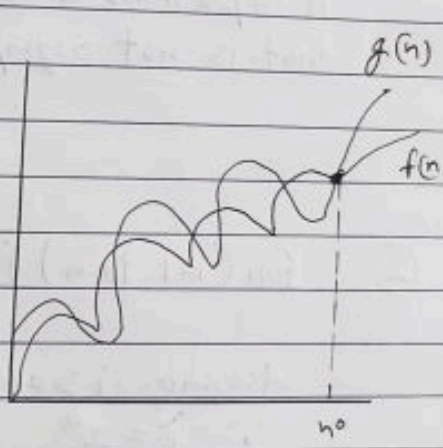
Date.
Page No.

Design and Analysis of Algorithm

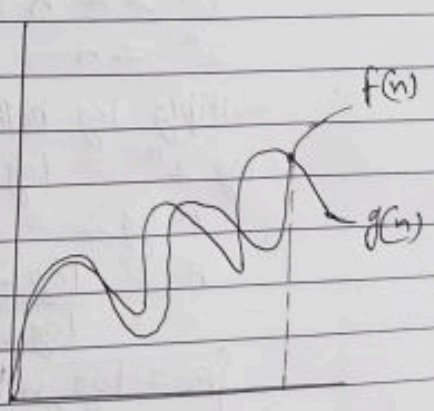
Assignment – 1

① Asymptotic notation is a mathematical notation used to describe the behavior of functions as their input approaches infinity.

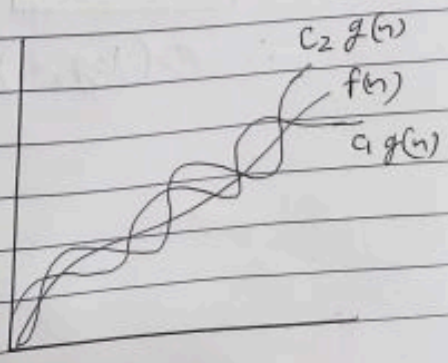
i) Big O Notation (O)
it represents the upper bound of an algorithm's growth rate
$f(n) = O(g(n))$
$g(n)$ is tight upper bound of $f(n)$
$f(n) \leq (g(n))$ $\forall n >= n^0$
and for some constant $c > 0$



ii) Big Omega Notation (Ω)
it represents the tight lower bound of an algorithms growth rate $f(n) = \Omega(g(n))$
$g(n)$ is tight lower bound of $f(n)$
Example :- if an algo has a time complexity of $\Omega(n)$ it means it means algo time grows at least linearly with size of input

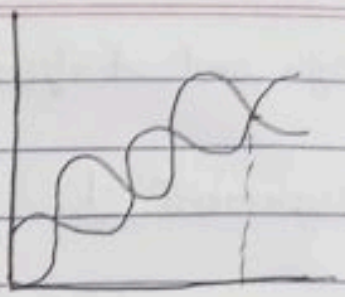


iii) Theta Θ Notation (Θ)
it represents both the upper and lower bound of an algo.
Example :- If an algo has time complexity $\Theta(n)$ it means algo runtime grow linearly with size of input neither fast nor slow

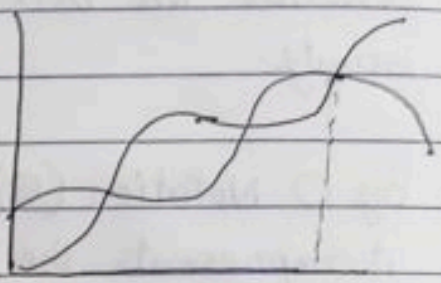


Date.
Page No.

iv) Small O notation (o)
it represents an upper bound that is not asymptotically tight



v) Small Omega notation (ω)
it represents a lower bound that is not asymptotically tight



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

i
1
1 x 2 = 2
2 x 2 = 2^2
2^2 x 2 = 2^3
2^3 x 2 = 2^4
⋮
2^k

Assume $i >= n$

$\because$ $i = 2^k$

$2^k >= n$

$2^k = n$

multiply log both side

$\log 2^k = \log n$

$k \log 2 = \log n$

$k = \frac{\log n}{\log 2}$

$\boxed{k = \log_2 n}$

$\therefore O(\log_2 n)$

(3)
$$\begin{cases} 3T(n-1) & \text{if } n>0 \\ 1 & \text{otherwise} \end{cases}$$

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

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

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

$T(n) = 3 \times 3^2 T(n-2)(n-1)$

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

After n steps :-

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

continue till $n-n=0$

$n-n=0$

$n=n$

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

$\boxed{T(n) = 3^n}$

Date.
Page No.

(5)

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

i	s
1	1+1 = 2
2	~~2+2=4~~ 1+1+2 = 4
3	1+1+2+3 = 7
4	1+1+2+3+4 = 11
.	.
.	.
K	

Assume $s > n$ → stoping condition

$1+1+2+3+4 \dots + K$

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

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

$\frac{K(K+1)}{2} > n-1$

$K^2 > n-1$

$K > \sqrt{n-1}$

$\therefore$ ~~a(sqrt)~~ $= \boxed{O(\sqrt{n})}$

Date.
Page No.

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

$i * i < n$

$i * i = n$

$i^2 = n$

$i = \sqrt{n}$

the loop runs until i reach upto $\sqrt{n}$

$\therefore$ T.C is $O(\sqrt{n})$

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

$n + (n \log n) + (n \times \log n \times \log n)$

$n + n\log n + (n \log n \times \log n)$

~~$n + 2n\log n$~~

$n + n\log n + n\log n^2$

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