Chandransh Saxena
11 CST

Page Date

Tutorial :- 1

Ans1> Asymptotic notation is used to describe the running time of an algorithm and how much time of an algorithm takes with a given input and when input is very large.

Types of notations -:

1) Big Oh notation, O → The notation O(n) is the formal way to express the upper bound of an algorithm's running time. It measure the worst case time complexity on the longest amount of time an algorithm can possibly take to complete.

Eg -: for function $f(n)$

$O(f(n)) = O\{g(n)\}$ $\forall\ c > 0, n > n_0$

$f(n) \leq c.g(n)$

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

2) Big Omega notation, Ω → The notation Ω(n) is the formal way to express the lower bound of an algorithm's running time. It measure the best case time complexity or the best amount of time an algorithm can possibly take to complete.

eg -: $\Theta(f(n)) = \Omega(g(n))$ $\forall\ c>0, n \geq n_0$

$g(n) \leq c.f(n)$

3) Theta notation, θ → This notation is formal way to express both lower bound and the upper bound of an algorithm's running time.

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

if $C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n)$

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

$C_1 > 0$ & $C_2 > 0$

Ans2) for(i=1 to n) // $O(\log(n))$

{

i=i*2; // $O(1)$

}

For i = 1, 2, 4, 8, 16 - - - 2^k this means (k) times as per this code, it will run till $2^k = n$ which means $k = \log n$ then

Complexity is $O(\log n)$

$\sum_{i=1}^{n} 1+2+4+8+ - - - - - +n$

$T_k = ar^{k-1} \Rightarrow 1 \times 2^{k-1}$, $n = 2^{k-1}$

$2n = 2^k$

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

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

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

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

Ans3) $T(n) = \{3T(n-1)$ if $n>0$, otherwise $1\}$

$T(n) = 3T(n-1)$ —— ①

Put $n = n-1$

$T(n-1) = 3T(n-2)$ —— ②

from ① to ②

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

$\Rightarrow 9T(n-2)$ —— ③

Put $n = n-2$ in (i)

$T(n) = 3(T(n-3))$ —— (4)

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

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

$\{k=3\}$

Put, $n-k=0$

$n=k$

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

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

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

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

$\{T(0)=1\}$

Ans 4) $T(n) = \begin{cases} 2T(n-1) & , n>0 \\ 1 & , n \leq 0 \end{cases}$

Using backward substitution,

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

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

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

⋮

$T(1) = 2T(0)$

$T(0) = 1$

Substituting value of $T(n-1)$ then $T(n-2)$ ---- till $T(1)$ in eq^n $T(n)$

we get,

$T(n) = 2^n \times T(0)$

$\because T(n) = 2^n \times 1$

$\Rightarrow O(2^n)$

Ans5→ $i = 1, 2, 3, 4, 5, 6 - - - - -$

$S = 1 + 3 + 6 + 10 + 15 + 21 + - - - + n$

$0 = 1 + 2 + 3 + 4 + - - - + n - T(n)$

$1 + 2 + 3 + 4 + - - - - k > n$

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

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

Ans6→

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

$i = 1, 2, 3, - - - n$

$i^2 = 1, 4, 8, - - - - n$

$i^2 <= n$ & $i <= \sqrt{n}$

k^{th} term, $t_k = a + (k-1)d$

$a = 1, d = 1$

$t_k <= \sqrt{n}$

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

$\sqrt{n} = k$

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

Ans7→

i	j	k
n/2	logn	logn
n/1	⋮	⋮
n	logn	logn

Page Date

$\frac{n+1}{2}^{th}$

$$\Rightarrow O(i*j*k) = O\left[\left(\frac{n+1}{2}\right) * \log n * \log n\right]$$

$$\Rightarrow O\left[\left(\frac{n+1}{2}\right) * (\log_2 n)^2\right]$$

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

Ans 8 →

$T(n) = T(n-3) + n^2$ —— (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)

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

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

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

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

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

$T(n) = 1 + 4^2 + 6^2 + \ldots + n^2$

$\Rightarrow n^2 + \ldots + 1$

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

Ans 9 →

i	j
1	n times
2	$1+3+5+---+n$ times

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

$a=1, d=2$

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

$\frac{n-1}{2} = k-1 \Rightarrow k = \frac{n-1}{2}+1 \Rightarrow k = \frac{n+1}{2}$ // No of terms.

For $i=2$, $j = \frac{n+1}{2}$ times

for $i=3$, $j = 1+4+7+---+n$ times

$n = 1+(k-1)d$

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

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

$k = \frac{n+2}{3}$ // No. of terms.

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

$T(n) = n + \frac{n+1}{2} + \frac{n+2}{3} + --- + \frac{n+k-1}{k}$ = n term

$\because \sum_{i=1}^{n} \frac{n+k-1}{k} \Rightarrow \frac{\sum_{i=1}^{n} n + \sum_{i=1}^{n} k - \sum 1}{k}$

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

$$\Rightarrow \frac{n^2 + n + 2nk - 2n}{2k}$$

After remaring constant term and lower terms

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

Ans10) $n^k = O(c^n)$

as $n^k \leq a.c^n$

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

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^k \leq a(2)$

$n_0 = 1$ & $c = 2$