Tutorial-1

Ques 1

Asymptotic Notations :- It give us an idea about how good a given algorithm is, as compared to some other algorithms.

There are 3 types of widely used asymptotic notations

1) Big O (O)
2) Big Omega (Ω)
3) Big theta (Θ)

i) **Big O Notations :-** This notification defines an upper bound of an algorithm, it bounds a function only from above.

ii) **Omega Notations :-** Just as Big O notations provides an asymptotic upper bound on a function, Ω notations provides an asymptotic lower bound.

iii) **theta Notation :-** This notation bounds a function from above & below, so it defines exact asymptotic notation.

eg :- $f(n) = \sum_{i=1}^{n} i \times 2^i$ $\longrightarrow$ $T(n) = \Omega(2^n)$

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

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

Ques 2 Time complexity of — `for(i=1 to n){ i = i*2; }`

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

$t_k = ar^{k-1}$

$n = 2^{k-1}$

$\log_2 n = k-1$

$k = \log_2 n + 1$

$O(k) = O(\log_2 n + 1)$

$\Rightarrow T(n) = O(\log_2 n)$ — Ans

Ques3 $T(n) = \{ 3T(n-1) \text{ if } n>0, \text{ otherwise } 1 \}$

$T(0) = 1$

$T(n) = 3T(n-1)$ — (1)

put $n = n-1$ in eqn (1)

$T(n-1) = 3T(n-2)$ — (2)

$T(n-1) = 3^2 T(n-2)$ — (3)

put $n = n-2$ in eqn (3)

$T(n-2) = 3T(n-3)$ — (4)

put in eqn (3)

$T(n) = 3^3 T(n-3)$ — (5)

for some constant K

$T(n) = 3^K T(n-K)$ — (6)

put $n-K=0 \Rightarrow K=n$

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

$\rightarrow \boxed{T(n) = O(3^n)}$ — Ans

Ques4 $T(n) = \{ 2T(n-1) - 1 \text{ if } n>0, \text{ otherwise } 1 \}$

$T(n) = 2T(n-1) - 1$ — (1)

$T(n-1) = 2T(n-2) - 1$ (put $n = n-1$ in eqn (1))

$T(n) = 2^2 T(n-2) - 3$ — (2)

$T(n-2) = 2T(n-3) - 1$ — (3) (put $n = n-2$ in eqn (1))

$T(n) = 8T(n-3) - 4 - 2 - 1$ — (4)

$T(n) = 8T(n-3) - 7$ — (4)

for some constant K, $T(n) = 2^K T(n-K) - 2^{K-1} - 2^{K-2} - \dots - 2^0$

put $n = K = 0 \Rightarrow n = K$

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

$a = 2^{n-1}$, $r = +1/2$, $S = \frac{2^n[1/2 - 1]}{1/2 - 1} = 2^n[2^{-n} - 1]$

$T(n) = 2^n - 2^n[2^{-n} - 1] = 2^n[1 - 2^{-n} + 1] = 2^n[2 - 2^{-n}]$

$\rightarrow$ **$T(n) = O(2^n)$** — Ans

Ques 5

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

i =	1	2	3	4	5	6	
s =	1	3	6	10		T_n	①
s =	1	3	6	10	----	$T_{n-1} + T_n$	②

Subtract eqn ② from eqn ①

$0 = 1 + 2 + 3 + 4 + \text{----} \; n - T_n$

$T_K = 1 + 2 + 3 + 4 \text{----} K$

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

for K iterations

$1 + 2 + 3 + \text{----} + K <= n$

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

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

$O(K^2) <= n$, $K = O(\sqrt{n})$

$T(n) = O(\sqrt{n})$ — Ans

Ques 6

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

$\because i^2 <= n \Rightarrow i \leq \sqrt{n}$

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

$\sum_{i=1}^{n} = 1 + 2 + 3 + 4 \text{---} + \sqrt{n}$

$T(n) = \frac{\sqrt{n}(\sqrt{n}+1)}{2} = \frac{\sqrt{n} + n}{2}$, **$T(n) = O(n)$** Ans

Ques 7

```
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++;
```

for $k = k^2$

$k = 1, 2, 4, 8 \quad \cdots \quad n$

$a=1, k=2 \Rightarrow k = \log_2 4$

i	j	k
$n/2$	$\log n$	$\log n * \log n$
$n/2+1$	$\log n$	$\log n * \log n$
⋮	⋮	⋮
n	$\log n$	$\log n * \log n$

$T(n) = O[n/2 * \log n * \log n * \log n]$

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

Ques 8

```
function (int n) {
    if (n==1) return ——————— O(1)
    for (i=1 to n) { ——————— O(n)
        for (i=1 to n) { ——————— O(n)
            }          printf("*"); ——————— O(1)
    }
    function (n-3); }    // T(n/3)
```

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

Using master's method

$a=1, b=3, f(n) = n^2$

$c = \log_3 1 = 0$

$n^c = 1 > n^2$

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

Ques 9

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

for $i=1$, $j = 1, 2, 3, \ldots$ — $n = n$

for $i=2$, $j = 1, 3, 5 \ldots$ — $n/2 = n/2$

for $i=3$, $j = 1, 4, 7 \ldots$ — $n/3 = n/3$

$= 1$

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

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

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

$$\Rightarrow \sum_{j=n}^{1} n(\log n)$$

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

Ques 10

Assume that $K >= 1$ & $c > 1$ are constants

Relation b/w n^K & c^n is $n^K = O(c^n)$

as $n^K \leq a c^n$

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

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^K \leq a \cdot 2^1$

$\boxed{n_0 = 1 \quad \& \quad c = 2}$ Ans