Kunal Bandooni
CE
2014413

PAGE No.
DATE: / / 202

★ Tutorial - 1

Q1. What do you understand by Asymptotic Notation.

Ans 1. Asymptotic Notation → These are mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value.

Example :-

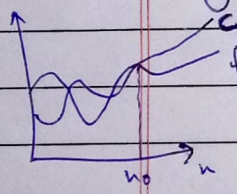
⇒ Bubble Sort

- if input array is already sorted, T.C. = O(n)
- but if it is unsorted, T.C. = O(n²).

Types of Asymptotic notation :-

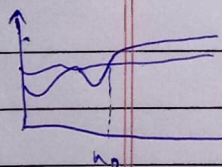
1. Big - O Notation (O)
2. Omega Notation (Ω)
3. Theta Notation (Θ)

1. Big - O → It represents upper bound of running time of an algo.
∴ It gives worst-time complexity



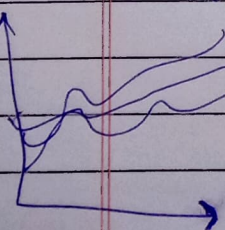
$f(n) = O(g(n))$

2. Omega → It represents lower bound of an algorithm.
∴ It provides best case complexity



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

3. Theta → It represent lower & upper bound of algo
∴ It provides average case complexity



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

Ans2. for (i=1 to n) { i = i*2; }

$i = 1, 2, 4, 8, 16$
$2^0 \quad 2^1 \quad 2^2 \quad 2^3 \quad 2^4$

GP $\Rightarrow a = 1$, $r = \frac{2}{1} = \frac{4}{2} = \frac{8}{4} = 2$

$t^k = ar^{k-1}$
$n = 1(2)^{k-1}$
$n = \frac{2^k}{2}$
$2n = 2^k$
$k = \log_2 2n$
$k = \log_2 2 + \log_2 n$
$k = 1 + \log_2 n$

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

Ans3. $T(n) = \{3T(n-1)$ if $n>0$ otherwise $1\}$
$T(0) = 1$
$T(1) = 3T(0) = 3$
$T(2) = 3T(1) = 3 \times 3 = 3^2$
$T(3) = 3T(2) = 3^3 \quad \ldots \quad 3^n$

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

Ans4. $T(n) = \{2T(n-1)-1$ if $n>0$, otherwise $1\}$

$T(0) = 1$

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

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

$T(3) = 2T(2) - 1 = 2-1 = 1$

...

$\therefore T(n) = O(1^n) = O(1)$

Ans5.

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

$i = 1, s = 1 = 1$

$i = 2, s = 3 = 1+2$

$i = 3, s = 6 = 3+3$

$i = 4, s = 10 = 6+4$

$i = 5, s = 15 = 10+5$

...

$\therefore TC = O(n)$

Ans6.

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

$i = 1, i*i = 1$

$i = 2, i*i = 4$

$i = 3, i*i = 9$

$i = 4, i*i = 16$

...

$i = \sqrt{n}, i*i = n$

$\therefore TC = O(\log n)$

Ans7.

```
for(i = n/2; i<=n; i++)                    — O(n/2)
    for(j = 1; j<=n; j = j*2)              — O(log n)
        for(k = 1; k<=n; k = k*2)          — O(log n)
            c++;
```

$\therefore TC = O(\frac{n}{2} \log n \log n)$

$TC = O(n \log n \log n)$

Ans8.
```
function (int n){
    if (n==1) return;
    for (i=1 to n)                    - O(n)
        for (j=1 to n)                - O(n)
            printf("*");

    function (n-3);                   - O(n)
}
```

TC = O(n * n * n)
TC = O(n3)

Ans9.
```
void function(int n) {
    for (i=1 to n)                    - O(n)
        for (j=1; j<=n; j=j+i)        - O(n)
            printf("*");
}
```

TC = O(n*n)
TC = O(n²)

Ans10. n^k & c^n — Asymptotic relationship b/w these
$k >= 1$ & $c > 1$ — constants
Find out the value of c & no for which relation holds.

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