Tutorial - 1

1. What do you understand by Asymptotic notations. Define diff. Asymptotic notation with example.

Asymptotic means tending to infinity.
Asymptotic notations are used to represent the complexity of an algorithm when the input is very large.
There are 5 Asymptotic notations

1. Big Oh (O) Notation

Big-oh(O) notation gives an upper bound (tight) for a function $f(x)$ within constant factor

$$f(n) = O\,g(n)$$
$$\text{iff } f(n) \leq c * g(n)$$
$$\forall\, n \geq n_0$$

Ex. $f(n) = 2n^3 + 1$
$\rightarrow O(n^3)$

2. Big Omega (Ω) Notation

Big-Omega (Ω) Notation gives tight or strict lower bound of function $f(n)$

$$f(n) \geq c * g(n)$$
$$\forall\, n \geq n_0$$

3. Theta (θ) Notation

Theta (θ) Notation gives both upper and lower bound.
$f(n) = \theta(g(n))$ iff
$$c_1 g(n) \leq f(n) \leq c_2 g(n) \qquad \forall\, n \geq \max(n_1, n_2)$$

4. Small 'o' Notation

It gives the upper bound of a function $f(n)$

$$f(n) = o\,g(n)$$

$$f(n) < c * g(n)$$

$$\forall\ n > n_0$$

$$\forall\ c > 0$$

5. Small 'w' Notation

It gives the lower bound of a function $f(n)$

$$f(n) = \omega\, g(n)$$

iff

$$f(n) > c * g(n)$$

$$\forall\ n > n_0$$

$$\forall\ c > 0$$

Q2. Time complexity of-

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

$i \rightarrow$ 1 2 4 8 16 ---- n

$$k = 1 * (2^k - 1)$$

$$2n = 2^k$$

$$\log_2 2 + \log_2 n = k \log_2 2$$

$$1 + \log n = k$$

$$T.C. = O(\log_2 n)$$

Q3. $T(n) = \{ 3T(n-1)$ if $n > 0$, otherwise $1 \}$

→ $T(n) = 3T(n-1)$ — (1)

Putting $n = n-1$ in eq^n (1)

$T(n-1) = 3T(n-2)$ — (2)

Putting (2) in (1)

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

→ $T(n) = 9T(n-2)$ — (3)

Putting $n = n-2$ in eq^n 1

$T(n-2) = 3T(n-3)$ — (4)

Putting (4) in (3)

→ $T(n) = 9[3T(n-3)]$

$T(n) = 27T(n-3)$ — (5)

$T(n) = 3^k T(n-k)$ — (6)

$T(1) = 1$

$n-k = 1$

$n = k$

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

$= 3^n$

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

Q4 $T(n) = \{2T(n-1)-1$ if $n>0$, otherwise $1\}$

→ $T(n) = 2T(n-1)-1$ — (1)

Using Backward Substitution

$T(n-1) = 2T(n-2)-1$ — (2)

Putting (2) in (1)

→ $T(n) = 2[2\{T(n-2)-1]-1$ — (3)
$= 4T(n-2)-3$

$T(n-2) = 2T(n-3)-1$ — (4)

Putting (4) in (3)

$T(n) = 2[2\{2T(n-3)-1\}-1]-1$ — (5)

$= 2[2\{4T(n-3)-3]-1$

→ $= 8T(n-3)-7$

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

$n-k = 1$

$n = k$

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

$= 1$

$T(n) = O(1)$

T.C = O(1)

Q5 What should be the time complexity of -

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

while loop terminates if : $1+2+3 \text{ ------ } +k$

$= k[k+1)/2 > n$

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

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

Q6

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

$i^2 <= n$

$i <= \sqrt{n}$

$i = 1, 2, 3, 4 \text{ ------- } \sqrt{n}$

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

$$= \frac{n+\sqrt{n}}{2}$$

$T(n) = O(n)$

Q7

```
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

i	j	k
$n/2$	1	1
$n/2+1$	2	2
$n/2+2$	4	4
⋮	⋮	⋮
n	n	n
$\left(\frac{n}{2}+1\right)$	$(\log_2 n+1)$	$(\log_2 n+1)$

$n = 1(2)^{k-1}$

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

$(k = \log_2 n + 1)$

$$T.C = O\left(\left(\frac{n}{2}+1\right)(\log_2 n+1)(\log_2 n+1)\right)$$

$$= O(n \log_2 n \log_2 n)$$

$$\boxed{T.C. = O(n \log^2 n)}$$

Q8

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

Using Master's method

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

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

$$c = \log_3 1 = 0$$

$$n^0 \geq 1$$

$$f(n) > 1$$

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

Q9.
```
void function (int n) {
    for (i = 1 to n) {
        for (j = 1; j <= n; j = j + i)
            printf("*");
    }
}
```

for $i = 1 \Rightarrow$ $j = 1, 2, 3, 4, \ldots\ldots\ldots n$ $\qquad n$

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

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

$\vdots$

for $i = n \Rightarrow$ $j = 1 \ldots\ldots\ldots n$ $\qquad n$

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

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

$$\boxed{T.C = O(n \log n)}$$

Q10 For functions n^k and c^n, what is asymptotic notations between these functions?
Assume that k>=1 and c>1 are constants. Find out the value of c and no of relation holds.

Given: n^k, c^n

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

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

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

for $n_0 = 1$

$c = 2$

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

$n_0 = 1$ and $c = 2$