Q1) Asymptotic notation

Himanshi Rana.
20, CST SPL 2

They help us to find the complexity of an algo when input is very large.

1. Big O(O). – It gives the worst case complexity.
2. Omega notation – It gives the best case complexity
3. Theta notation – It gives the average case complexity.

Example –

Bubble sort algorithm has O(n) time complexity in best case and ($O(n^2)$) time complexity in worst case and $O(n^2)$ in average case.

Q2).

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

$i = 1, 2, 4, 8, \ldots n \rightarrow$ G.P.

$a_k = a r^{n-1}$ $\quad a = 1, r = 2$

$a_k = 1 - 2^{k-1}$

$n = 2^{k-1}$

$\log_2 n = k - 1$

$k = 1 + \log_2 n$

$\therefore T(n) = O(\log_2 n + 1) = O(\log n)$. Ans

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

$T(0) = 1$

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

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

$T(n-1) = \cancel{3t} \; 3T(n-2) \rightarrow (2)$

put (2) in (1).

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

put $n = n-2$ in eq (1).

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

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

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

let $n-k = 0$

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

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

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

$T(n) = 2T(n-1) - 1 \longrightarrow (1)$.

$T(0) = 1$

put $n = n-1$

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

$\cancel{4T(n-2) - 2 - 1} \; \cancel{2^2 T(n-2) - 2 - 1}$.

put (2) in 1

$T(n) = 4T(n-2) - 2 - 1 \rightarrow (3)$

Let $n = n-2$

$T(n-2) = 2T(n-3) - 1 \rightarrow (4)$

from (3) & (4)

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

$T(n) = 8T(n-3)-4-2-1$

$T(n) = 2^k T(n-k) - \cancel{2k}\, 2^{k-1} - 2^{k-2} - 2^{k-3} \dots 2^0$

Let $n-k=0$

$n=k$

$T(n) = 2^n T(n-n) - 2^{n-1} - 2^{n-2} - 2^{n-3} \dots 2^0$

$T(n) = 2^n T(0) - 2^{n-1} - 2^{n-2} - 2^{n-3} \dots 2^0$

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

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

$\{\because 2^{n-1} + 2^{n-2} \dots + 2^0 = 2^n - 1\}$

$T(n) = 1$ Ans

$T(n) = O(1)$

Q5)

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

$i=1$	$s=1$	
$i=2$	$s=3$	$s=1+2$
$i=3$	$s=6$	$s=1+2+3$
$i=4$	$s=10$	$s=1+2+3+4$

$s = 1+2+3+4 \dots + k = \frac{k(k+1)}{2} > n$

$s = \frac{k^2+k}{2} > n$

$k > \sqrt{n}$

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

Q6 void function (int n)

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

$i = 1, 2, 3, \ldots n$

$i^2 = 1, 2^2, 3^2 \ldots n^2$

$i^2 <= n$

$\Rightarrow i <= \sqrt{n}$

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

$a = 1, d = 1$

$a_k <= \sqrt{n}$

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

$\sqrt{n} = k$

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

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++;
            }
        }
    }
}
```

i	j	k
$n/2$	$\log n$	$(\log n)^2$
$\frac{n+1}{2}$	$\log_2 n$	$(\log_2 n)^2$
⋮	⋮	⋮
n	$\log n$	$(\log_2 n)^2$

$\frac{n}{2}+1$ times.

$O(i * k) = O\left(\left(\frac{n}{2}+1\right) * (\log n)^2\right)$

$T(n) = O(n(\log n)^2)$ Ans

Q8

```
function (int n)
{
    if (n==1)
        return;
    for (j=1 to n)
    {
        for (j=1 to n)
            print("*");
    }
    function(n-3);
}
```

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

$T(1) = 1$

put $n = n-3$ in eq (1)

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

put $T(n-3)$ in (1)

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

put $n = n-6$ in (1)

$T(n-6) = T(n-9) + (n-6)^2 \longrightarrow (4)$

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

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

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

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

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

$T(n) = 6n^2 + k$

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

Q7 void function(int n)

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

$i = 1, \quad j = 1, 2, 3, 4 \dots$ n times

$i = 2, \quad j = 1, 3, 5, 7 \dots$ $n/2$ times.

$i = 3, \quad j = 1, 4, 7, 11 \dots$ $n/3$ times.

$i = n \quad j = 1 \dots$ 1 times

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

$$\sum_{j=1}^{n} n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots \frac{1}{n}\right)$$

$$= n(\log n)$$

$$T(n) = (n \log n)$$

$$T(n) = O(n \log n)$$ Ans

Q10

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

as $n^k <= 2c^n \quad \forall\, n \geq n_0$

for $n_0 = 1$

$c = 2$

$1^k \leq a_2$

$n_0 = 1 \;,\; c = 2$