1) Time Complexity & space complexity

2) What is Big O.

3) How to calculate Big O

4) Time limit Exceed (TLE)

[ ] $[a, b] \to$ inclusion

( ) $(a, b) \to$ Exclusion

$[a, b)$

$[3, 10] = 8$ 3 4 5 6 7 8 9 10

$[a, b] = b - a + 1$

$(a, b) = b - a - 1$ ($\downarrow -2$)

2. Arithmetic Progression

$4, 7, 10, 13, 16, 19, \ldots$

3 3 3 3 3

$a, a+d, a+2d, a+3d, a+4d, \ldots$

$a \to$ first term

$d \to$ common diff

$N \to$ no of terms.

Sum of first N terms =

$$a + (a+d) + (a+2d) + \cdots + (a+(n-1)d)$$

$$= na + d + 2d + 3d + \cdots (n-1)d$$

$$= na + d(1+2+3+\cdots (n-1))$$

$$= na + d\,\frac{n(n-1)}{2}$$

$$\boxed{S_n = \frac{n}{2}\left[2a + (n-1)d\right]}$$

2. Geometric Progression (GP)

$$\overbrace{5, 10}^{2}, \overbrace{10, 20}^{2}, \ldots \quad 5, 10, 20, 40, 80, 160 \ldots\ldots$$

$$a, ar, ar^2, ar^3 \ldots\ldots$$

$a \to$ first term

$r \to$ common ratio

$N \to$ no of terms.

$$\boxed{S_N = \frac{a(r^n - 1)}{r - 1}} \qquad r > 1$$

$5, 10, 20, 40$ $\qquad a = 5 \quad N = 4$

$r = 2$

$$S_4 = \frac{5(2^4-1)}{2-1} = \frac{5(15)}{1} = 75$$

4. log Basics

$\log_x a$: No of time we need to divide a by x, till we get 1

$$\log_2 2^{10} = 10$$

$$\log_{10} 10^{12} = 12$$

$$\log_a a^x = x$$

$$a^x \downarrow a^{x-1} \downarrow a^{x-2} \vdots a^0$$

Q.

```
int fn(N) {
    s = 0
    for( i=1 ; i≤ N; i++)
    {      s = s+i
    return s
}
```

$i : [1, N]$

No of iterations $= N-1+1$
$= N$

O(N)

Q. void fn (int N, int M)

```
{
    for( i=1 ; i<= N ; i++ )
    {   print(i)

    for( j=1 ; j<= m ; j++ )
    {   print(j)
}
```

i : [1, N]

N − 1 + 1

= N

j : [1, m]

m − 1 + 1

= m

No of iterations = N + m

O(N + m)

Q. int fn (N)

```
{
    s = 0
    for( i=1 ; i ≤ N ; i=i+2 )
    {    s = s+i

    return s.
}
```

N = 9

[1, 2, 3, 4, 5, 6, 7, 8, 9]

$$\frac{N+1}{2}$$

No of iterations = $\frac{N+1}{2}$

$= \frac{N}{2} + \frac{1}{2}$

O(N)

N = 6

[1, 2, 3, 4, 5, 6]

$$\frac{6+1}{2} = \frac{7}{2} = 3$$

Q. int fn(N)

```
{
    s = 0
    for( j = 0 ; j ≤ 100 ; j++ )
    {   s = s + j
    return s.
```

$j : [0, 100]$

$100 - 0 + 1$

No of iterations, $= 101\, N^0$

$O(N^0) = O(1)$

Q. void fn(N)

```
{
    s = 0
    for( i = 1 ; i*i ≤ N ; i++ )
    {   s = s + a
```

$i = 1$

$i \times i \le N$

$i^2 \le N$

$i \le \sqrt{N}$

$i : [1, \sqrt{N}]$

$\sqrt{N} - 1 + 1$

No of iterations $= \sqrt{N}$

$O(\sqrt{N})$

Q. void fn(N)

```
{
    while (N > 1)
    {
        N = N/2
    }
}
```

$N \rightarrow \frac{N}{2} \rightarrow \frac{N}{4} \rightarrow \frac{N}{8} \rightarrow \dots \rightarrow 1$

No of iterations $= \lfloor \log_2 N \rfloor$

$10 \rightarrow 5 \rightarrow 2 \rightarrow 1$

$\log_2 10 = 3.$

iteration no	value of N	
1	$\frac{N}{2}$	$\frac{N}{2^1}$
2	$\frac{N}{4}$	$\frac{N}{2^2}$
3	$\frac{N}{8}$	$\frac{N}{2^3}$
⋮		
K	1	$\frac{N}{2^K}$

$\frac{N}{2^K} = 1$

$N = 2^K$

$\log_2 N = \log_2 2^K$

$\log_2 N = K$

$O(\log_2 N)$

Q. Void fn(N)

```
{
    s = 0
    for( j=0 ; j<=N ; j = j*2)
    { s = s+j
}
```

$j = 0 \rightarrow 0 \rightarrow 0 \rightarrow 0 \ldots$

No of iterations = infinite

Q. Void fn(N)

```
{
    s = 0
    for( j=1 ; j<=N ; j = j*2)
    { s = s+j
}
```

$1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \ldots \rightarrow N$

N = 32

$32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

/2

$\log_2 N$

*2

$N = 10$

$1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow \cancel{16}$

iteration	value	
1	2	2^1
2	4	2^2
3	8	2^3
4	16	2^4
⋮		
k	N	2^K

$2^K = N$

$\log_2 2^K = \log_2 N$

$K = \log_2 N$

$O(\log_2 N)$

Q. void fn(N)

```
{
    for (i=1 ; i<=10; i++)
    {
        for(j=1; j ≤ N; j++)
        {
            print( i+j)
        }
    }
}
```

i	j	iteration of inner loop
1	[1:N]	N +
2	[1:N]	N +
3	[1:N]	N +
⋮		⋮

⋮		⋮
10	[1:N]	N

No of iteration = $10N$

$O(N)$

Q.

```
void fn(N)
{
    for(i=1; i<=N; i++)
    {
        for(j=1; j≤N; j++)
        {
            print(i+j)
        }
    }
}
```

i	j	iteration of inner loop
1	[1:N]	N +
2	[1:N]	N +
3	[1:N]	N +
⋮		⋮ +
N	[1:N]	N

No of iterations = $N \times N = N^2$

$O(N^2)$

Q.

```
void fn(N)
{
    for(i=1; i≤N; i++)
    {
        for(j=1; j<=N; j=j*2)
        {
            print(i+j)
        }
    }
}
```

$\log_2 N$

i	j	No of inner loop iteration
1	$1 \longrightarrow N$	$\log_2 N$ +
2	$1 \longrightarrow N$	$\log_2 N$ +
3	$1 \longrightarrow N$	$\log_2 N$ +
⋮		
N	$1 \longrightarrow N$	$\log_2 N$

No of iterations $= N \times \log_2 N$

$O(N \log N)$

Q. Void fn (N)

```
{
    for( i=1 ; i ≤ 2^N ; i++)
    {   print(i)
}
```

$i : [1, 2^N]$

$= 2^N - 1 + 1$

No of iteration $= 2^N$ $O(2^N)$

Q. void fn (N)

```
{
    for( i=1 ; i ≤ N ; i++)
    {
        for( j=1 ; j <= 2^i ; j++)
        {   print( i+j)
    }
}
```

i	$j : [1 : 2^i]$	No of iteration of inner loop
1	$[1, 2^1]$	2
		$+$
2	$[1, 2^2]$	2^2
		$+$
3	$[1, 2^3]$	2^3
		$+$
$\vdots$		$\vdots$
		$+$
N	$[1, 2^N]$	2^N

No of iterations $= 2 + 2^2 + 2^3 + 2^4 + \ldots 2^N$

GP

$a = 2$

$r = 2$

N terms.

$$S_N = \frac{a(r^n - 1)}{r - 1} = \frac{2(2^N - 1)}{2 - 1}$$

$$= 2(2^N - 1)$$

$$= 2^{N+1} - 2$$

$$= 2 \cdot 2^N - 2$$

$$O(2^N)$$

How to calculate Big O Notation from no of iterations.

1. Neglect all lower order terms
2. Neglect all constant coefficent

iteration = $4N^2 + 3N + 1$

$4N^2 \longrightarrow N^2$

Big O

$O(N^2)$

Q. # of iterations = $4N^2 + 3N + 10^6$

$\longrightarrow N^2$ $\quad O(N^2)$

Q. $3N\sqrt{N} + 4\log N + 31 N\log N$.

$\longrightarrow O(N\sqrt{N})$

$N = 2^{32}$

$N\sqrt{N}$

$2^{32} \; 2^{16}$

2^{48}

$N \log N$

$2^{32} \; 32$

$2^{32} \; 2^{5}$

2^{37}

$K < \log N < \sqrt{N} < N < N\log N < N\sqrt{N} < N^2 < N^x < 2^N < N! < N^N$

Doubts

$N = 10^{18}$ # of iterations $= 10^9$

$i = 4$ $N = 16$

$i = n/i$

$\text{if}(i == \frac{N}{i})$

$i \times i = n$

$\text{if}(i \times i == N)$