Time Complexity 1

Good Evening

Today's Agenda

TC 2

- → Time complexity / Space complexity
- → Big O → What? Why? How?
- → TLE, Time Limit Exceeded?
- → Online compilers

TC 1 → No. of iterations → Quizzes (20 - 25)

Basic Maths

01. Sum of first N natural numbers = $\frac{n * (n+1)}{2}$

$$S_6 = 1 + 2 + 3 + 4 + 5 + 6 = 21$$

$$S_6 = \frac{6 * (6+1)}{2} = \frac{\overset{3}{\cancel{6}} * 7}{\cancel{2}} = 21$$

02. Elements in Range

Numbers in Range : [3 10] ⇒ 3, 4, 5, 6, 7, 8, 9, 10

⇓

8

Numbers in Range : [4 7] ⇒ 4, 5, 6, 7 ⇒ 4

Numbers in Range : [a b] ⇒ b - a + 1

AP → difference b/w two adjacent ele must always be same

Series → 2 4 8 16 32 . . .

$$\frac{4}{2} = \frac{8}{4} = \frac{16}{8} = \frac{32}{16}$$

Between two adjacent elements common ratio is same $\longrightarrow$ GP

Series $\rightarrow$ 2 10 50 250 ... $\longrightarrow$ GP

$$5 = \frac{10}{2} = \frac{50}{10} = \frac{250}{50}$$

Sum of first n terms in GP $= a\left\{\frac{r^n - 1}{r - 1}\right\}$

a = first term

r = common ration

n = no. of terms

Series $\rightarrow$ $2 + 4 + 8 + 16 + 32 = 62$

$a = 2$

$r = 2$

$n = 5$

$$S_5 = 2\left\{\frac{2^5 - 1}{2 - 1}\right\}$$

$$S_5 = 2\left\{\frac{2^5 - 1}{1}\right\}$$

$$S_5 = 2\{31\} = 62$$

log basics

$$\log_b^a = c \longleftrightarrow b^c = a$$

$$\log_3^{81} = 4 \longleftrightarrow 3^c = 81$$

$$3^c = 3^4$$

$$\log_2^{2^5} = 5$$

$$N = 2^K \longleftrightarrow K = \log_2^N$$

* **No. of iterations**

```
01. void fun (int n){
      int s=0
      for( int i=1 ; i<= 100 ; i++){
        print ("Hi");
      }
    }
```

$i \in [1 \;\; 100]$ (a = 1, b = 100) $= 100 - 1 + 1$

$= 100$ iterations

Big O → O(1)

02.
```
void fun (int n){
    int s=0
    for( int i=3; i≤50 ; i++){
        print ("Hi");
    }
}
```

a b = b - a + 1

i : [3 50] = 50 - 3 + 1

= 48 iterations

Big O → O(1)

07.
```
void fun (int n){
    int s=0
    for( int i=1 ; i*i≤n; i++){
        print ("Hi");
    }
}
```

i * i ≤ n → i² ≤ n → i ≤ √n

i = [1 √n] = √n iterations

Big O → O(√n)

03.
```
void fun (int n){
    int s=0
    for( int i=1 ; i≤ n ; i++){
        print ("Hi");
    }
}
```

i : [1 N] → N iterations

Big O → O(n)

04.
```
void fun (int n){
    int s=0
    for( int i=0; i<n ; i++){
        print ("Hi");
    }
}
```

$i : [0 \quad n-1]$

$\Rightarrow n-1-0+1$

$= n-1+1$

$\Rightarrow n$ iterations

05.
```
void fun (int n , int m){
    for( int i=1; i≤n ; i++){
        print ("Hi");
    }
    for( int i=1; i≤m; i++){
        print ("Hi");
    }
}
```

$i : [1 \quad n] \rightarrow n$ iterations

$i : [1 \quad M] \rightarrow M$ iterations

Total iterations = $N+M$

06.
```
void fun (int n){
    int s=0
    for( int i=1 ; i≤2^n ; i++){
        print ("Hi");
    }
}
```

→ $i : [1 \; 2^n] = 2^n$ iteration

$i = n$

08.
```
void fun (int n){
    int i=n
    while (i>1){
        print (Hi);
        i= i/2
    }
}
```

Iterations	Value of i after each iteration
After 1st	$i = \frac{n}{2} = \frac{n}{2^1}$
After 2nd	$i = \frac{i}{2} = \frac{n}{4} = \frac{n}{2^2}$
After 3rd	$i = \frac{i}{2} = \frac{n}{8} = \frac{n}{2^3}$
⋮	
After kth	$i = \frac{n}{2^k}$

Obs 1 → After k iteration = $i = \frac{n}{2^k}$

Obs 2 → when $i = 1$, the loop stops

$$\frac{n}{2^k} = 1$$

$$n = 2^k$$

Take $\log_2$ on both sides

$$\log_2 n = \log_2(2^k)$$

$$\log_2 n = k$$

08.
```
void fun (int n){
    int i=n
    while (i>0){
        print (Hi);
        i= i/2
    }
}
```

$i = \frac{n}{2^{k-1}}$ $\quad i=1$

$$\frac{n}{2^{k-1}} = 1 \Rightarrow n = 2^{k-1}$$

$$k-1 = \log_2 n$$

$$k = \log_2 n + 1$$

09.
```
void fun (int n){
    int s=0
    for( int i=0; i< n ;i=i*2){
        print ("Hi");
    }
}
```

$i=0$ $\quad 0<n \rightarrow$ Hi

$i=0*2$ $\quad 0<n \rightarrow$ Hi
$= 0$

$i=0*2$ $\quad 0<n \rightarrow$ Hi
$= 0$

⋮

Hi

Infinite iterations

10.

```
void fun (int n){
    int s=0
    for( int i=1 ; i< n ;i=i*2){
        print ("Hi");
    }
}
```

Iterations	value of i → 1 after the iteration
After 1st	i = i * 2 = 2 → 2^1
After 2nd	i = i * 2 = 4 → 2^2
After 3rd	i = i * 2 = 8 → 2^3
After 4th	i = i * 2 = 16 → 2^4
⋮	⋮
After k th	$1 = 2^k$

Obs 1 → After k^{th} iteration, $i = 2^k$

Obs 2 → Loop stops, $i = N$

$$2^k = N$$

$$k = \log_2 N$$

Note :-

$i = N$ & $i = i/2$ → No. of iteration = $\log_2 N$

$i = 1$ & $i = i * 2$ → No. of iterations = $\log_2 N$

Time for break → 10:02 → 10:12 pm

* Nested Loops

11.
```
void fun (int n)
{
    for (int i=1 ; i<=3 ; i++){
        for (int j=1 ; j<=4 ; j++){
            print ("Hi");
        }
    }
}
```

i	j	iterations
1	[1 4]	4 Hi
2	[1 4]	4 Hi
3	[1 4]	4 Hi
4 → stop		12 iterations

12
```
void fun (int n)
{
    for (int i=1 ; i≤10 ; i++){
        for (int j=1 ; j ≤N ; j++){
            print ("Hi");
        }
    }
}
```

i	j	iterations
1	[1 N]	N Hi
2	[1 N]	N Hi
3	[1 N]	N Hi
⋮		⋮
10	[1 N]	N Hi
11 → stop		10 N

13
```
void fun (int n)
{
    for (int i=1 ; i≤N ; i++){
        for (int j=1 ; j ≤N ; j++){
            print ("Hi");
        }
    }
}
```

No. of iteration = N^2

```
14 void fun (int n)
{
    for (int i=1 ; i≤N ; i++){
        for (int j=1 ; j <N ; j=j*2){
            print ("Hi");
        }
    }
}
```

i	j	iterations
1	[1 N-1)	log n
2	[1 N-1)	log n
3	[1 N-1)	log n
⋮	⋮	⋮
N	[1 N-1)	log n

n

N * log n iteration

```
15 void fun (int n)
{
    for (int i=0; i<N ; i++){
        for (int j=0 ; j≤i ; j++ ){
            print ("Hi");
        }
    }
}
```

i	j	iterations
0	[0 0]	1
1	[0 1]	2
2	[0 2]	3
.	.	.
.	.	.
.	.	.
(n-1)	[0 n-1]	n

Total no. of iteration = $1+2+3+\cdots+n$

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

*
```
void fun (int n)
{
    for (int i=1; i<=N; i++){
        for (int j=1; j<=i; j++ ){
            print ("Hi");
        }
    }
}
```

i	j	iterations
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
4	[1 4]	4
.	.	.
.	.	.
n	[1 n]	n

$n(n+1)/2$

```
16 void fun (int n)
   {
     for (int i=1 ; i<N ; i++){
       for (int j=1 ; j≤i ; j++ ){
         print ("Hi");
       }
     }
   }
```

i	j	iterations
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
⋮	⋮	⋮
(n-1)	[1 n-1]	n-1

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

$$S_x = \frac{x(x+1)}{2}$$

$$S_{n-1} = \frac{(n-1) * (n-x+x)}{2}$$

$$S_{n-1} = \frac{n * (n-1)}{2} \text{ iteration}$$

17.
```
void func(int n)
{
    for (int i=1 ; i<N ; i++){
        for (int j=1 ; j<= 2^i ; j++){
            print("Hi");
        }
    }
}
```

i	j	iterations
1	$[1 \quad 2^1]$	2^1
2	$[1 \quad 2^2]$	2^2
3	$[1 \quad 2^3]$	2^3
.		.
.		.
.		.
(n-1)	$[1 \quad 2^{n-1}]$	2^{n-1}

n ⟶ stop

Total iterations $= 2^1 + 2^2 + 2^3 + \cdots + 2^{n-1}$

$a = 2$

$r = 2$

no. of terms $= n-1$

$$S_n = a\left\{\frac{r^n - 1}{r-1}\right\} = 2\left\{\frac{2^{n-1} - 1}{2-1}\right\}$$

$$= 2\{2^{n-1} - 1\}$$

$$= 2 * 2^{n-1} - 2 = 2^n - 2 \text{ iterations}$$

* Comparing functions → for large values of N

$$\log n < \sqrt{N} < N < N\log N < N\sqrt{N} < N^2 < 2^n < n^n$$

Google for $N!$

High Order Terms

(a) $\log N$ — $N \log n$

(b) $2N$ — N^2

(c) $4N^2$ — 2^n

Big O : What? Why?

How to calculate Big O

(a) Calculate the no. of iterations

(b) Consider the higher order terms

(c) Neglect constant coefficient

	$f(n)$ → No. of iterations	HOT	Big O
01.	$3n^2 + 10n + 20$	$3n^2$	$O(n^2)$
02	$5n^2 + 10n^3 + 1$	$10n^3$	$O(n^3)$
03	$3n + 5\sqrt{n} + 2n\log n$	$2n\log n$	$O(n\log n)$

04.

```
void fun (int n){
    int s=0
    for( int i=1 ; i<= 10 ; i++){
        print ("Hi");
    }
}
```

10 iterations

constant

If the no. of iterations doesn't change on changing the input

$O(1)$

Doubt Session

$\log N < \sqrt{N}$

$\sqrt{N} = A$

$N = A^2$

Take $\log_2$ on both sides

$\log_2 N = \log_2(A^2)$

$K+1 = 2^n$

$K = 2^n - 1$

2^n 1

HOT = 2^n

Big O → $O(2^n)$

$j = 1$

```
(j=1 ; j<=N ; j=j*2) {
|
3
```

$j = N+1$

	iteration count	j
After	1	$2 \rightarrow 2^1$
After	2	$4 \rightarrow 2^2$
After	3	$8 \rightarrow 2^3$
	.	.
	.	.
After	K	2^k

Obs → $j = 2^k$ at the last iteration

$j = N+1$

$2^k = N+1$

$\log_2 2^k = \log_2 (N+1)$

$k = \log_2 (N+1)$

Big O → $O(\log_2 N)$

A & B

↓

count of mangoes

count of slices of mangoes

⇓

3 slices

1 Shake ⇒ 2 slices

A = 7 (B = 1) Output = 11

↓

21 slices + 1

22 slices

2 slices = 1 glass

1 slice = $\frac{1}{2}$ glass

22 slices = $\frac{1}{\cancel{2}} \times \overset{11}{\cancel{22}}$ glasses

= 11 glasses