Time Complexity

⇒ RULES FOR THIS CLASS

- → Get a notebook to write down.
- → If you are getting questions in any concept like what if we replace a Variable X with Y then what will be Ans? Note it down.
- → If by the end of class your's noted question is not answered then ask in Doubt Session.

AGENDA

- → Logs Basics + Iteration problems
- → Comparing iterations using graph.
- → Time complexity – Defination & Notations (Asymptotic Analysis – Big O)
- → TLE
- → Importance of Constraints

BASIC OF LOG

① $\log_2(64)$ = Power of 2 to get 64
= 6

② $\log_3(27) = 3$

③ $\log_5(25) = 2$

④ $\log_2(8) = 3$

GENERIC

$\log_b(a)$ = what should be pow of b to get a

Let, $\log_b(a) = c$

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

Eg:- ① $\log_2(10)$ = 3. something ≈ 3

② $\log_2(40)$ = 5. something ≈ 5

Note:- $2^K = N \Rightarrow \log_2 N = K$

And, $\log_2(2^6) = 6$

$\log_3(3^5) = 5$

$\log_a(a^N) = N$

Q) Divide N by 2 till it becomes 1

Ex:- N = 100

$100 \rightarrow 50 \rightarrow 25 \rightarrow 12 \rightarrow 6 \rightarrow 3 \rightarrow 1$

Ans = 6 times

N = 324

$324 \rightarrow 162 \rightarrow 81 \rightarrow 40 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 2 \rightarrow 1$

Ans = 8 times

Quiz 1 :- Divide 9 by 2 till it reaches 1

$9 \rightarrow 4 \rightarrow 2 \rightarrow 1$ Ans=3

GENEREAICALLY

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

After K iterations

$$\Rightarrow N \rightarrow \frac{N}{2^1} \rightarrow \frac{N}{2^2} \rightarrow \frac{N}{2^3} \rightarrow \ldots\ldots\ldots \rightarrow \frac{N}{2^K}$$

we can observe :-

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

from above relation,

$$\Rightarrow \boxed{\log_2 N = K}$$

Conclusion :- After $\log_2 N$ iterations N becomes 1.

Quiz 2 :- N = 27, How many time I need to divide by 2 to get 1

$\log_2(27)$ = 4. something

= 4

Leaven this, If we want to get 1 by dividing N, K times by 2 then the Ans of K is $\log_2(N)$

ITERATION QUESTIONS

Quiz 3 :- How many iterations will be there in this loop?

```
N > 0
i = N;

while (i > 1)
{
    i = i/2;
}
```

⇒ O(log N)

Explaination :- [N → 1], So iterations will be $\log_2(N)$

Quiz 4 :- How many iterations will be there in this loop?

```
for (i = 1; i < N; i = i * 2)
{
    ....
}
```

⇒ O(log N)

Ans :- [1 N] → $\log_2 N$

Quiz 5 :- How many iterations will be there in this loop?

```
N >= 0

for (i=0 ; i<=N ; i = i*2)
{
    ..............
}
```

⇒ Infinite

Answer = Infinite loop.

Quiz 6 :- How many iterations will be there in this loop?

```
for ( i=1 ; i <=10 ; i++) {
    for (j= 1;  j <=N; j++) {
        / ..... /
    }
}
```

⇒ 10 N

O(N)

Let, learn a new way.

i	j	iterations
1	[1 N]	N-1+1 ⇒ N
2	[1 N]	N
3	[1 N]	N
⋮	⋮	⋮
10	[1 N]	N
	Sum	10 N

10 times

Quiz 7 :- How many iterations will be there in this loop?

```
for ( i=1 ; i <= N ; i++) {
    for (j= 1;  j <=N; j++) {
        | ..... |
    }
}
```

$\Rightarrow N^2$

$O(N^2)$

i	j	iterations
1	[1 N]	N - 1 + 1 $\Rightarrow$ N
2	[1 N]	N
3	[1 N]	N
⋮	⋮	⋮
N	[1 N]	N

N terms

Sum N * N

Quiz 8 :- How many iterations will be there in this loop?

```
for ( i=1 ; i <= N ; i++) {
    for (j= 1;  j <=N; j=j*2) {
        | ..... |
    }
}
```

$\Rightarrow N \log N$

$\Rightarrow O(N \log N)$

i	j	iterations
1	[1 N]	$\log_2 N$
2	[1 N]	$\log_2 N$
⋮	⋮	
N	[1 N]	$\log_2 N$

N terms

$N * \log_2 N$

Quiz :- How many iterations will be there in this loop?

```
for ( i=1 ; i <= 4 ; i++) {
    for (j= 1;  j <= i ; j ++) {
        // print(i+j);
    }
}
```

=> (10)

=> O(1)

i	j	iterations
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
4	[1 4]	4
	Total	10

Quiz 10 :- How many iterations will be there in this loop?

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

i	j	iterations
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
⋮		⋮
N	[1 N]	N
	Total	$\frac{N(N+1)}{2}$

$\Rightarrow \frac{N(N+1)}{2}$

$\Rightarrow \frac{N^2}{2} + \frac{N}{2}$

$\Rightarrow \frac{N^2}{2}$

$\Rightarrow O(N^2)$

Quiz 11 :- How many iterations will be there in this loop?

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

$\Rightarrow 2(2^N - 1)$

$\Rightarrow 2.2^N - 2$

$\Rightarrow 2.2^N$

$\Rightarrow O(2^N)$

i	j	iterations
1	[1 2^1]	2^1
2	[1 2^2]	2^2
3	[1 2^3]	2^3
⋮	⋮	⋮
N	[1 2^N]	2^N
	Total	$2(2^N-1)$

} N terms

Here, $a = 2^1 = 2$

$r = \frac{2^2}{2^1} = 2$

$n = N$

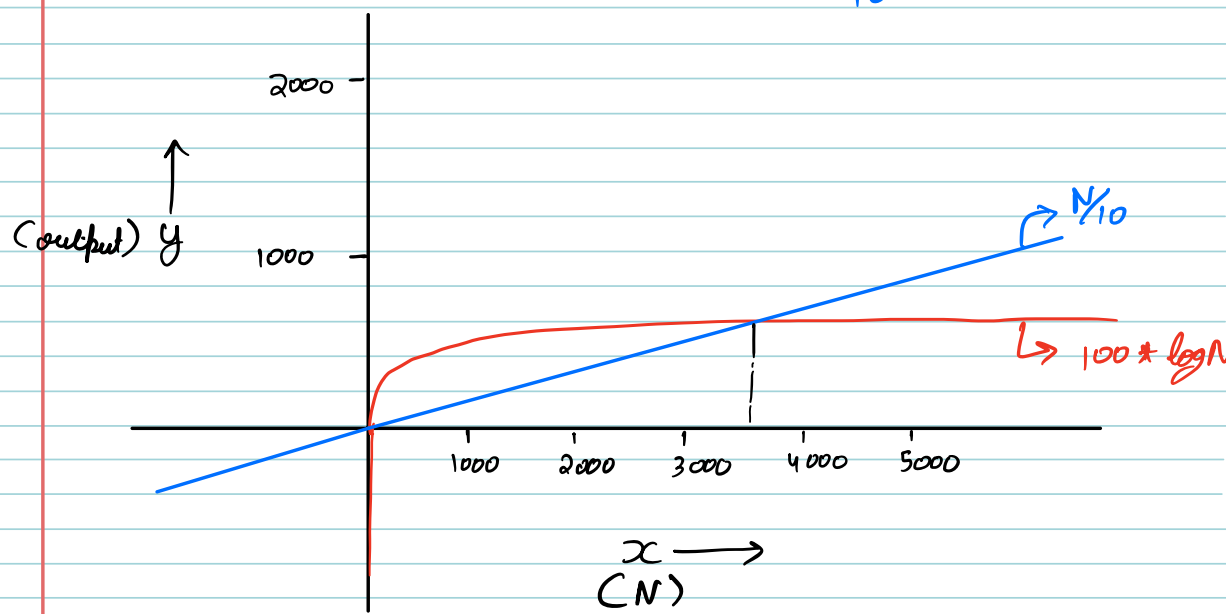
Sum of GP $= \frac{a(r^n-1)}{r-1}$

$= 2(2^N-1)$

Break :— 10:31 —— 10:39

Compare two different Algorithms

	No. of Iterations
Algo 1	$100 * \log(N)$
Algo 2	$\frac{N}{10}$



So $N < 3500$, Algo 2 is performing better

$N > 3500$, Algo 1 is performing better

In Real World, the Data is very large.

- → India Vs Math → 18M views
- → Baby Shark Video → 2888M views

Hence, we can say since the data is very large the less I will perform better.

ASYMPTOTIC ANALYSIS → (Big O)

* We use this Analysis to compare two Algo when input is huge.
* We use this Analysis to compare two Algos performance with larger inputs

↳ How to compute Big O

- → Calculate the no. of iterations
- → Ignore Lower order terms.
- → Ignore Constant terms.

Examples :-

Algo 1 ⇒ $100 * \log N$ ⇒ $O(\log N)$

Algo 2 ⇒ $\frac{N}{10}$ ⇒ $O(N)$

Examples :- $4N^5 + 3N^2 + 1$

⇒ $4N^5$

⇒ $O(N^5)$

*** V. Imp

COMPARISION ORDER

$\log(N) < \text{Sqrt}(N) < N < N\log(N) < N\sqrt{N} < N^2 < N^3 < 2^N < N! < N^N$

Ex = N = 36, the the order will be

$$5 < 6 < 36 < 36*5 < 36*6 < 36*36 < 36*36*36 < 2^{36} < 36! < 36^{36}$$

Quiz 12 :- $F(N) = 4N + 3N\log(N) + 1$

$= 3N\log N$

$= O(N\log N)$

Quiz 13 :- $F(N) = 4N\log N + 3N\sqrt{N} + 10^6$

$= 3N\sqrt{N}$

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

Questions :- why do we need lower order terms.

Let, $N^2 + 10N$ iterations

N	total iterations	contri of lower order terms
10	200	100 ≈ 50%
100	$10^4 + 10^3$	10^3 ≈ <9%
10000	$10^8 + 10^5$	10^5 ≈ 0.1%

Conclusion :- with increasing value of N, the contri of lower order terms decrease

drastically.

Question :- why do we need to neglect constant terms.

chetna (Algo 1)	Praveen (Algo 2)	winner
$10 \log_2 N$	N	chetna
$100 \log_2 N$	N	chetna
$9 * N$	N^2	chetna
$10 * N$	$N^2/10$	chetna
$N + \log_2 N$	$\log_2 N$	Praveen

Issues in Big O Notations

Issues 1 :-

		Algo 1	Algo 2
		$10^3 N$	N^2
	B(O)	O(N)	O(N²)
①	N = 10	10^4	10^2
②	N = 100	10^5	10^4
③	N = 1000	10^6	16^6
④	N = 10000	10^7	10^8

Conclusion :- Big (O) notations are not official for smaller values.

Issue 2

Code 1 :-

```
for ( i=1; i<=N; i++){
    if( i%2 !=0){
        c=c+1;
    }
}
```

Iterations

N

↓

O(N)

Code 2 :-

```
for (i=1; i<=N; i+=2){
    c=c+1;
}
```

Iterations

N/2

↓

O(N)

Conclusion :- Acc to Big O these two codes are same but we know Code2 will work better.

TIME LIMIT EXCEEDED

Venkata → Amazon → Contest

1st question → Code → Submit → TLE

(2 iterations)

* By time Venkata Succeed Interview will be over.

Question :- Can we some how assess the logic ability before even testing

↳ online Editor [Standard]

↳ 1 GHz machine

↳ 10^9 instructions in 1 Sec

↳ Generally after 1 Sec your will get TLE

Instructions :- Its a Single operation like function call, addition, multiplication etc.

Code :-

```
int CountFactors (int N) {
    int c=0;
    for ( i=1; i<=N; i++) {
        if (N%i==0) {
            c++;
        }
    }
```

Roughly 6 instructions

```
    return c;
}
```

Approximation 1 :- In a small code, generally
1 iteration ≤ 10 instructions
∴ 10^8 iteration ≤ 10^9 instruction

Approximation 2 :- In a large code, generally
1 iteration = 100 instructions
10^7 iterations = 10^9 instruction

Conclusion :- if our code is having 10^8 or 10^7 iterations, then only it will be Submitted in 1 Sec.

How to Solve problem

- Read the **Question** and **Constraints** carefully.
- Formulate an **Idea** or **Logic**.
- Verify the **Correctness** of the Logic.
- Mentally develop a **Pseudocode** or rough **Idea of Loops**.
- Determine the **Time Complexity** based on the Pseudocode.
- Assess if the time complexity is feasible and won't result in **Time Limit Exceeded (TLE)** errors.
- **Re-evaluate** the **Idea/Logic** if the time constraints are not met; otherwise, proceed.
- **Code** the idea if it is deemed feasible.

How to Read Constraints

Eg: Constraints

consider Worst Case

$1 <= N <= 10^5$

T.C.

$O(N^2)$

$O(N^3)$

$O(N \log N)$

Eg Constraints

$1 <= N <= 10^6$

T.C

$O(N^3 \log N)$

$O(N^2)$

$O(N \log N)$

$O(N)$

$10^6 \log(10^6) \leftarrow$ May or May not $\leftarrow 2 \times 10^7$

Eg Constraints

$1 < N < 100$

T.C

$O(N^3)$

$\hookrightarrow (10^2)^3$

$\hookrightarrow 10^6$

Eg:- Constraints

$1 <= N <= 20$

T.C.

$O(2^N)$

$\hookrightarrow (2^{20}) \approx 10^6$

NEXT SESSION

- Space Complexity
- Arrays.