Time Complexity

1. log basics + iterations
2. Comparing iterations using graph
3. Time Complexity — Big O
4. TLE
5. Imp. of constraints

Logarithms

log is inverse of exponential function

$\log_b(a)$ → To what power we need to raise to 'b' s.t. we get 'a'.

$$b^c = a \Rightarrow c = \log_b(a)$$

$\log_2 64 = 6$ → $2^c = 64$ ⇒ $\boxed{c = 6}$

$\log_3 27 = 3$

$\log_5 25 = 2$ → $5^c = 25$ $\boxed{c = 2}$

$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$

$\log_2 10 \Rightarrow 2^c = 10$

$2^3 = 8$

$2^4 = 16$ $\quad 3 < c < 4$

$c = 3.$~~xxxx~~

$\log_2 15 \Rightarrow 2^c = 15$

$2^3 = 8$

$2^4 = 16$

$c = 3.$~~xxxx~~

Always take floor in case of decimal answer

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

$\log_2(2^6) = 6$

$\log_3(3^5) = 5$

$\Rightarrow \log_a(a^N) = N$

$\log_2(3^5)$

$2^c = 3^5 = 243$

$c = 8.\ldots$

Question:

Given N, how many times do we need to divide it by 2, until it reaches 1. $N > 0$

$N = 100$

$100 \xrightarrow{/2} 50 \xrightarrow{/2} 25 \xrightarrow{/2} 12 \xrightarrow{/2} 6 \xrightarrow{/2} 3 \xrightarrow{/2} 1$

count = 6

$N = 9$

$9 \xrightarrow[1]{/2} 4 \xrightarrow[2]{/2} 2 \xrightarrow[3]{/2} 1$ count = 3

$N \xrightarrow{1^{st}} N/2 \xrightarrow{2^{nd}} N/4 \xrightarrow{3^{rd}} N/8 \rightarrow \dots \rightarrow 1$

$N/2^0 \xrightarrow{1^{st}} N/2^1 \xrightarrow{2^{nd}} N/2^2 \xrightarrow{3^{rd}} N/2^3 \rightarrow \dots \xrightarrow{K^{th}} N/2^K$

$N/2^K = 1$

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

times division to get 1

Quiz: $N = 27$ times to get to 1

$K = \log_2 27$

$K = 4$

$2^4 = 16$

$2^5 = 32$

Quiz

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

$i = N \xrightarrow{1^{st}} N/2 \xrightarrow{2^{nd}} N/4 \ldots\ldots \xrightarrow{K^{th}} N/2^K$

$N/2^K = 1 \Rightarrow K = \log N$ or $\log_2 N$ $O(\log N)$

Quiz 4

```
for(i=1 ; i<N; i=i*2){
    ...
}
3
```

$i=1 \xrightarrow[1^{st}]{\times 2} i=2 \xrightarrow[2^{nd}]{\times 2} i=2^2 \xrightarrow[3^{rd}]{\times 2} i=2^3 \rightarrow \ldots\ldots \rightarrow i=2^K$ (K^{th})

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

$O(\log N)$

$N = 64 \Rightarrow \log_2 N = 6$

$N/2 = 32$

Quiz 5

$N >= 0$

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

$i = 0 \xrightarrow{\times 2} i = 0 \xrightarrow{\times 2} i = 0 \ldots\ldots \xrightarrow{\times 2}{}_{\infty \text{ times}} 0$

loop will never stop → Infinite

Quiz 6

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

i	j	iterations
1	[1,N]	N
2	[1,N]	N
3	.	.
.	.	.
.	.	.
.	.	.
10	[1,N]	N

$= 10N$

$\Rightarrow 10N$

$O(N)$

Quiz 7

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

$\Rightarrow N \times N$

$= N^2$

$O(N^2)$

i	j	iteration
1	[1,N]	N
2	[1,N]	N
.		.
.		.
.		.
N	[1,N]	N

$= N \times N$

Quiz 8

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

$= N \log N$

$O(N \log N)$

i	j	iteration
1	[1,2,4,8..N]	logN
2		logN
.		.
.		.
.		.
N		logN

$= N \log N$

Quiz 9

```
for (i=1; i<=4; ++i) {
    for (j=1; j<=i; ++j) {
        ...
```

3^7

10 iterations

$O(1)$

i	j	iterations
1	[1,1]	1 +
2	[1,2]	2 +
3	[1,3]	3 +
4	[1,4]	4
		= 10

Quiz 10

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

3^3

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

$O(N^2)$

i	j	iterations
1	[1,1]	1 +
2	[1,2]	2 +
3	[1,3]	3 +
.	.	.
.	.	.
.	.	.
N	[1,N]	N +

$1+2+3+\ldots+N$

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

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

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

Quiz 1!

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

i	j : [1, 2^i]	iterations
1	[1, 2^1)	2 +
2	[1, 2^2]	4 +
3	[1, 2^3]	8 +
.	.	.
.	.	.
.	.	.
N	[1, 2^N]	2^N +

$$2 + 4 + 8 + \dots + 2^N$$

eg $r = 2$, $a = 2$, N terms

$$\text{sum} = a\left(\frac{r^n - 1}{r - 1}\right) = 2\left(\frac{2^N - 1}{2 - 1}\right)$$

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

$$O(2^N)$$

Algo 1 — Samar

$100 \log_2 N$

Algo 2 — Khushboo

$N/10$

$N = 8 \Rightarrow$ algo 1 = $\log \log 8 = 300$

algo 2 = $8/10 = 0.8$ ✓

for small input ($N <= 3500$) , Algo 2 is better

for large input ($N > 3500$) , Algo 1 is better ✓

Ind v Pak viewers → 18M

Baby Shark video → 2.8B

Asymptotic Analysis of algo Big (O)

↳ it works for sufficiently large input

steps to calculate Big O

1. Calculate iterations based on input size
2. Ignore lower Order terms
3. Ignore constant coeffients

Algo 1 → $100 \log N$ $O(\log_2 N)$

Algo 2 → $N/10$ $O(N)$

iterations :

$4N^2 + 3N + 1$ → lower order terms

const. coff.

$\Rightarrow O(N^2)$

$\sqrt{N}\log N$

$1 < \log N < \sqrt{N} < N < N\log N < N\sqrt{N} < N^2 < N^3 < 2^N < N! < N^N$

	$\log N$	$\sqrt{N}$
$N = 4$	2	2
16	4	4
64	6	8
256	8	16

$\Rightarrow \log N < \sqrt{N}$

iterations : $4N^2 + 3N + 6\sqrt{N} + 9\log N + 10$

$\Rightarrow O(N^2)$

$f(n)$: $4N + 3N\log N + 1$

$\Rightarrow O(N\log N)$

$f(n): 4N\log N + 3N\sqrt{N} + 10^6$

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

Why ignore lower order terms?

N	iterations N^2+10N	lower oder $10N$	% $\frac{10N}{N^2+10N} \times 100$
$= 10$	$100+100=200$	100	$\frac{100}{200} = 50\%$
100	$10000+1000$	1000	$= \frac{10^3}{10^4+10^3} \approx 9\%$
1000	10^6+10^4	10^4	$\frac{10^4}{10^6+10^4} \approx 0.1\%$

Why ignore const. coff.

$10\log N$ ✓ N

$100\log N$ ✓ N

Issues with Big O

Algo1 $\rightarrow 10^3 N \rightarrow O(N)$

Algo2 $\rightarrow N^2 \rightarrow O(N^2)$

$N = 10, 100, 1000$ Algo 2 is better

iteration 1 = N → O(N)

iteration 2 = N/2 → O(N)

↓ always better

Quizly

```
for (i=N; i>0; i=i/2){
    for (j=1; j<=i; ++j){
        ...
    }
}
```

i	j:[1,i]	iterations
N	[1,N]	N +
N/2	[1,N/2]	N/2 +
N/4		N/4 +
⋮		⋮ +
1	[1,1]	1

$N/2^0 + N/2^1 + N/2^2 + \cdots + N/2^K$ $\quad N/2^K = 1$

$K = \log N$

$a = N,\ r = 1/2,\ n = \log N$

$$\text{sum} = a\left(\frac{r^n - 1}{r - 1}\right) \Rightarrow a\left(\frac{1 - r^n}{1 - r}\right)$$

$$= N\left(\frac{1 - (1/2)^{\log N}}{1 - 1/2}\right)$$

$$= N\left(\frac{1 - 1/2^{\log_2 N}}{1/2}\right)$$

$$= 2N\left(1 - 1/2^{\log N}\right)$$

$$= 2N\left(1 - \frac{1}{N}\right) = \frac{2N(N-1)}{N}$$

$$= 2(N-1)$$

$$= O(N)$$

$$\frac{1}{2^{\log_2 N}}$$

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

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

$$\log_2 a = \log_2 N$$

Time limit Exceeded

Online IDE $\rightarrow$ 10^9 instructions per sec

```
for (i=0; i<n; ++i) {
    s = s+i;
}
```

+1 , +1×n , +1 , +1×n

= 2 + 2n

3 iterations → 10 instruction

1 iteration → 100 instructions

1 sec → $10^7 - 10^8$ iterations

Importance of constraints

$1 <= N <= 10^5$

$O(N^3)$ $(10^5)^3 = 10^{15}$ ✗

$O(N^2)$ $(10^5)^2 = 10^{10}$ ✗

$O(N \log N)$ $10^5 \log 10^5 \approx 20 \times 10^5$ ✓✓

$10^3 \approx 2^{10} \rightarrow \log 2^{10} = 10$

$10^6 = 2^{20}$ $= 20$

$1 <= N <= 100$

$O(N^3)$ $(100)^3 = 10^6$ ✓