Agenda for today!

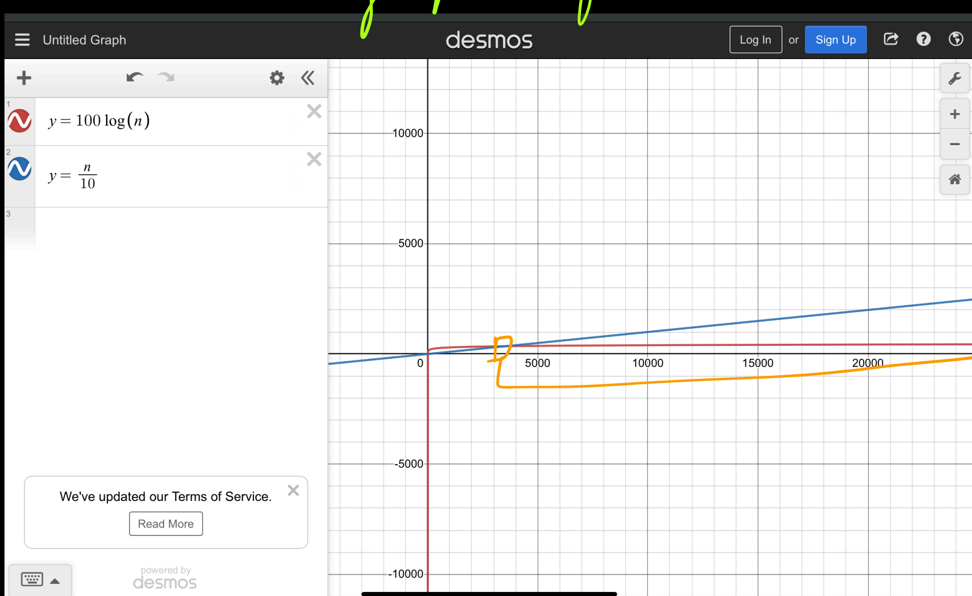
- → Comparing iterations using graph ✓
- → Time Complexity - Def. and Notations (Asymptotic Analysis - Big O) ✓
- → Log basics + Iteration Problems ✓
- → Space Complexity
- → TLE
- → Importance of Constraints

⇒ 20 min

Two algorithms developed by Vijay & Hariom to solve problem.

Algo	No. of iterations
Vijay	$100 * \log(N)$
Hariom	$N/10$

Let's draw graphs for this



→ 3500

For small input ($N <= 3500$) => Hariom algo performs better (Blue Line)

For large input ($N > 3500$) => Vijay's algo performs better (Red Line)

In today's world data is huge

→ Indian Vs Pak match viewship was 18 millions.

→ Baby Shark vied has 2.8 views.

We use Asymptotic Analysis to estimate the performance of an Algorith when input is large.

Asymptotic Analysis of Algorithms : Big (O) Notations

Analysis to get perforce of Algorithms for large inputs.

Calculation of Big (O)

Steps :

→ Calculate iterations based on input size.

→ Ignore lower order terms

→ Ignore constant coefficients

Ex: $100 * \log(N) \Rightarrow O(\log(N))$

$N/10 \Rightarrow O(N)$

$N^2 + N + 10 \Rightarrow$ Order $= 2$

No. of iteration: $4N^2 + 3N + 1$

Step 2: $4N^2 + 3N + 1$

Step 3: $4N^2$

Big O: $O(N^2)$

$\log_2(N) < \text{sqrt}(N) < N < N \log N < N\,\text{sqrt}(N) < N^2 < N^3 < 2^{N} < N! < N^{N}$

$N = 36$

$5 < 6 < 36 < 36 * 5 < 36 \cdot 6 < 36^2 < 36^3 < 2^{36} < 36! < 36^{36}$

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

$O(F(N)) = O(N \log(N))$

Trick: Try to put a large value and compare which term gives higher value.

Quiz 2: $4N\log(N) + 3N\,Sqrt(N) + 10^6$

$O(F(N)) = O(N\,Sqrt(N))$

Q Why do we neglect lower order terms?

$N^2 + 10N$

N	Total Iteration: $N^2 + 10N$	lower Order term $= 10N$	% of 10N in total iterations $(10N/(N^2+10N)) * 100$
10	$10^2 + 10*10 = 200$	$10*10 = 100$	$\frac{100}{200} \times 100 = 50\%$
100	$100^2 + 10*100 = 10^4 + 10^3$	$10*100 = 10^3$	$\frac{10^3}{10^4+10^3} \times 100 =$ close to 9%
10000	$(10^4)^2 + 10*10^4 = 10^8 + 10^5$	$10^4 * 10 = 10^5$	$\frac{10^5}{10^8+10^5} \times 100 =$ close to 0.1%

Conclusion: We can say that as the Input Size increases, contribution of Lower order Terms decreases

Why do we neglect constant coefficients

which algo do you choose => Algo with lesser iters. for large input

Mohan's Algo	Maddala's Algo	Winner for Larger Input
$10 * \log_2 N$	N	Mohan
$100 * \log_2 N$	N	Mohan
$9 * N$	N^2	Mohan
$10 * N$	$N^2/10$	Mohan
$N * \log_2 N$	$100 * N$	Maddala

$10 \underbrace{\log 10^4}$ $\quad 10^4 \quad 10 * 1000 \quad \frac{(1000)^2}{10}$

$1300 \quad 10000 \quad = 10^4 \quad = \frac{10^6}{10} = 10^5$

$100 * \underbrace{\log 10^6}_{20} \quad 10^6$

$10^4 * \underbrace{\log 10^4}_{15} \quad 100 * 10^4$

$2000 \quad 10^6 \quad 150000 \quad 100 \times 10^4 = 10^6$

$\shortparallel$

$2 * 10^3$

Try 10^6

$10^6 * \log 10^6 \quad 100 \times 10^6 = 10^8$

20×10^6

2×10^7

Conclusion: there is no effect of constant coefficients for higher values.

Issues with Big (O)

: Non - deterministic nature of Big (O)

Issue 1:

Alok => $10^3 N$: Big O: $O(N)$

Rakshit => N^2 : Big O: $O(N^2)$

Input Size	Alok's Algo	Rakshit's Algo	Better ?
$N = 10$	10^4	10^2	Rakshit
$N = 100$	10^5	10^4	Rakshit
$N = 10^3$	10^6	10^6	Equal
$N = 10^3+1$	$10^3 * (10^3+1)$	$(10^3+1) * (10^3+1)$	Alok
$N = 10^4$	10^7	10^8	Alok

Big O completely focuses on very high values.

Sometimes that high value is very high.

Issue 2:

Suraj's Code:

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

Iterations: N

Big O: O(N)

Akshay's Code:

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

Iterations: N/2

Big O: O(N)

In both, Big O is O(N) but we know second code is better.

when 2 algorithms have same Big O values then we are not capable of identify which is better.

Basics of Logarithm

Q What is in the meaning of Log ?

Logarithm is the inverse of exponential function.

Q How to read the statement " $\log_b(a)$ " ?

To what power we need to raise b, such that we get a.

$$b^{\boxed{?}} = a \longrightarrow \log_b a$$

Eg: 1) $\log_2(64)$ $\quad 2^{?} = 64$

$= 6 \quad 2^6 = 64$

2) $\log_3(27)$ $\quad 3^{?} = 27$

$= 3 \quad 3^3 = 27$

3) $\log_5(25)$ $\quad 5^{?} = 25$

$= 2 \quad 5^2 = 25$

4) $\log_2(32) = 5$

Floor $\Rightarrow$ $\text{Floor}(3) = 3$

$\text{Floor}(3.1) = 3$

$\text{Floor}(5.6) = 5$

$\text{Floor}(6.9999999) = 6$

Find floor of log values.

$\log_2(10) = 3$

$2^{?} = 10$

$2^3 = 8$ $\qquad$ $2^4 = 16$

3.xxx

$\log_2(40) = 5$

Some very basic property of log:

$\log_2(2^{\wedge}6) = 6$

$\boxed{2^{?} = 2^{6}} = ? = 6$

$\log_5(5^{20}) = 20$ $\qquad$ $5^{?} = 5^{20}$

$$\boxed{\log_a (a^N) = N}$$

$$2^K = N$$

Take log base 2 on both sides

$$\log 2^K = \log N$$

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

Only do integer division.

Quiz 3: How many we divide 9 by 2 till it reaches 1 ?

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

Quiz 3: How many we divide 27 by 2 till it reaches 1 ?

$$27 \xrightarrow[1]{} 13 \xrightarrow[2]{} 6 \xrightarrow[3]{} 3 \xrightarrow[4]{} 1$$

How many times we divide N by 2 till it reaches 1 ?

$$N \longrightarrow N/2 \longrightarrow N/4 \longrightarrow N/8 \cdots\cdots\cdots 1$$

$$\frac{N}{2^0} \longrightarrow \frac{N}{2^1} \longrightarrow \frac{N}{2^2} \longrightarrow \frac{N}{2^3} \cdots\cdots\cdots \frac{N}{2^K}$$

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

$$N = 2^K$$

$$K = \log_2 N$$

No. of steps : K steps $= \log_2 N$

Quiz 5: No. of iterations :

```
N > 0
i = N
while (i > 1) {
    i = i/2;
}
```

$i = N$
$i = N/2$
$i = N/4$
$i = N/8$
$\vdots$
$i = 1$

No. of iteration $= \log_2 N$

Quiz 6: No. of iterations :

$N >= 0$

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

$i = 0$
$i = 0$
$i = 0$
$i = 0$
$\vdots$
$i = 0$

No. of iterations : Infinite

Quiz 7: No. of iterations :

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

$i = 1$
$i = 2$
$i = 4$
$i = 8$
$\vdots$
$i = N$

No. of iterations : $\log_2(N)$

$$1 \xrightarrow{*2} 2 \xrightarrow{*2} 4 \xrightarrow{*2} 8 \;\cdots\cdots \xrightarrow{*2} N$$

$$\xleftarrow{/2} \quad \xleftarrow{/2} \quad \xleftarrow{/2} \quad \xleftarrow{/2} \quad \xleftarrow{/2}$$

How many steps to get 1 from N while dividing by 2 ? $\log_2 N$

Quiz 8: Find no. of iterations.

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

i	$j : [1 \quad N]$	# of iterations
1	$[1 \quad N]$	N
2	$[1 \quad N]$	N
⋮		
10	$[1 \quad N]$	N
		$10N$

Quiz 9:

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

i	$j : [1 \quad N]$	# of iterations
1	$[1 \quad N]$	N
2	$[1 \quad N]$	N
⋮		
N	$[1 \quad N]$	N
		$N * N$

Multiplying for loop is working here but will not work always.

Quiz 10:

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

i	j : [1 N] *2	# of iterations
1	1 2 4 ...... N	$\log_2 N$
2	"	$\log_2 N$
⋮	"	⋮
⋮	"	⋮
N	"	
		$N \log_2 N$

Quiz 11:

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

i	j	# iter
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
4	[1 4]	4
		10

Quiz 12:

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

i	j	# iter
1	[1 1]	1
2	[1 2]	2
3	[1 3]	3
4	[1 4]	4
5	[1 5]	
⋮	⋮	
N	[1 N]	N

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

Quiz 13:

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

i	j : [1 2^i]	# of iterations.
1	[1 2]	2
2	[1 4]	2^2
3	[1 8]	2^3
⋮		
N	[1 2^N]	2^N

GP

$a = 2, \ r = 2, \ N = N$

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

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

Quiz 14:

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

i	j	# of iteration
N	[1 N]	N
N/2	[1 N/2]	N/2
N/4	[1 N/4]	N/4
⋮		
1	[1 1]	1

$$N + \frac{N}{2} + \frac{N}{4} + \frac{N}{8} + \cdots\cdots 1$$

$$\frac{N}{2^0} + \frac{N}{2^1} + \frac{N}{2^2} + \frac{N}{2^3} + \cdots\cdots \frac{N}{2^K}$$

$$\frac{N}{2^K} = 1 \Rightarrow N = 2^K \Rightarrow K = \log_2 N$$

$$N\left(\frac{1}{2^0} + \frac{1}{2^1} + \frac{1}{2^2} + \frac{1}{2^3} \cdots\cdots \frac{1}{2^K}\right)$$

$[0, K] = K - 0 + 1 = K + 1$

GP: $a = 1 \quad r = 1/2 \quad$ # terms $= K + 1$

$$\frac{a(1 - r^N)}{1 - r} = \frac{1(1 - (1/2)^{K+1})}{1 - 1/2} = \frac{\left(1 - \frac{1}{2^{K+1}}\right)}{1/2}$$

$$= 2 * \left(1 - \frac{1}{2^K * 2}\right)$$

$(2^{K+1} = 2 * 2^K)$

$$2^K = 2^{\boxed{\log_2 N}} = N \qquad 2^{?} = N$$

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

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

No. of iteration: $N * \left(\frac{1}{2^0} + \frac{1}{2^1} \cdots \cdots \frac{1}{2^K}\right)$

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

$$= 2N - 1$$

Take away: $N + \frac{N}{2} + \frac{N}{4} \cdots \cdots 1 = 2N - 1$

$$K = \log_2 N$$

$$2^K = N$$

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