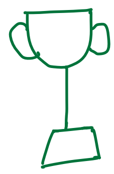
Is execution time a correct measure to compare algos ?

arr[] = {3, 1, 5, 2}

↓

{1, 2, 3, 5} ←



Abhishek [Speedboat]

10 s

Macbook pro

10 s

Python → C++

6.5 s

Mt Everest

Aakash [Sorteasy]

15 s

Windows → macbook pro.

7 s

C++

7 s

Volcano.

Execution time is not a good measure compare algorithms

- ↳ H/W
- ↳ Language
- ↳ Place
- ⋮

JS → C++ X

How to compare algorithms ?

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

i : [1 N]

Iterations : N

Iterations is a good measures to compare algorithms.

Rajiv

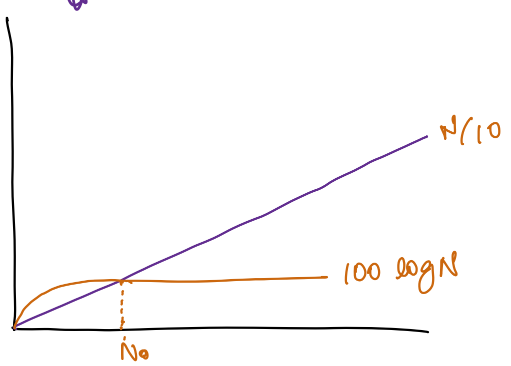
$100 \log N$

Rahul

$\frac{N}{10}$

$N <= 3550$ Rahul is performing better

$N > 3550$ Rajiv is performing better.



Hotstar	Despacito	Pingu
3 Cr	1 B	74 30 000
3×10^7	7×10^9	

$N \to \infty$

Asymptotic Analysis

performance of algorithm for very large input size tends to ∞.

- Big O [worst case]
- Omega Ω [Best Case]
- Theta θ [Average Case]

Read : Ref.

Big O

Steps to find Big O

1. ~~Calc #iterations~~
2. ~~Ignore lower order terms~~
3. ~~Remove contants from higher order term~~

$100 \log N \rightarrow O(\log N)$ Better

$\frac{N}{10} \rightarrow O(N)$

why to ignore lower order terms

$\boxed{N^2 + 10N}$

contribution of lower order term to the iterations.

	$N^2 + 10N$		$N^2 + 10N$	
$N = 10^2$	10^4	10^3	$10^4 + 10^3$	$\frac{10^3}{10^4 + 10^3} \times 100 = 9\%$
$N = 10^4$	10^8	10^5	$10^8 + 10^5$	$\frac{10^5}{10^8 + 10^5} \times 100 = 0.1\%$
$N = 10^5$	10^{10}	10^6	$10^{10} + 10^6$	$\frac{10^6}{10^{10} + 10^6} \times 100 = 0.01\%$

Home $\xrightarrow{384400 \text{ km}}$ Moon

Home $\xrightarrow{x}$ Airport $\xrightarrow{x}$ USA Airport $\xrightarrow{x}$ Space X $\xrightarrow{x}$ Moon

Why to remove the constants

Ketan

$10 \log_2 N$

$10^2 \log_2 N$

$10^4 \log_2 N$

$10^4 N + 10^6$

Wamika

N

N

N

N^2

Ketan wins ** .

Issues

Rafiqul

$\underbrace{10^3 N}$

$O(N)$

Vrushali

$\underbrace{N^2}$

$O(N^2)$

Can I claim Rafiqul is always better?

	$10^3 N$	N^2	
$N = 10$	10^4	10^2	Algo 2 is better.
$N = 10^2$	10^5	10^4	Algo2 is better
$N = 10^3$	10^6	10^6	Both are same
$N = 10^3 + 1$	$10^3 \times (10^3 + 1)$	$(10^3+1)(10^3+1)$	Algo 1 is better
:			

⋮

#1 Does not give a correct picture for smaller values of N.

Rafiqul wins **.

Another issue

$\frac{N^2 + 10N}{N^2}$ $\boxed{O(N^2)}$

#Iterations

$\frac{100N^2}{100N^2}$ $\boxed{O(N^2)}$

Space Complexity.

int : 4 B
long : 8 B
double : 8 B

```
void func(N) {
    int a,b,c
    long d
    double e
    print(a+b+c+d)
}
```

$3 \times 4 + 8 + 8$

SC: 28 Bytes.

O(1)

No matter how much N ↑, the space used by algo remains same.

```
void func(N) {
    int a,b,c
    long d
    double e
    int arr[N]
}
```

4 B + 3×4 B + 8 B + 8 B + 4×N B

$32 + 4 \times N = O(N)$ SC

```
void func(N){
    int a,b,c
    long d
    double w
    int arr[N]
    int mat[N][N]
}
```

$4\times4 + 8 + 8 + 4\times N + N^2\times 4$

$O(N^2)$

```
for( i=0; i<=N; i++){
    int x
}
```

Q.

```
                 4xN        4
int sum( int arr[], int N) {
       4
    int s=0
           4
    for( i=0; i< N ; i++){
        s = s+ arr[i]
    }
    return s
}
```

$4N + 12$

i/p space (given)

extra/ auxiliary space.

$= 12$

$= O(1)$

```
                  Nx4        4
int pref( int arr[], int N) {
    int sum[N] Nx4
    sum[0] = arr[0]
             4
    for(int i=1; i<N; i++){
        sum [i] = sum [i-1] + arr [i]
    }
```

i/p space

arr[N]	N	sum[N]	i
4xN	4	4xN	4

auxiliary

$4\times N + 4$

Extra space $\boxed{O(N)}$

}

```
bool func(int arr[], int N, int K) {
    for (int i=0; i<N; i++) {
        if (arr[i] == K) {
            return true
        }
    }
    return false.
}
```

K=4

4	1	3	2	7

Chandan.

Best case

$\boxed{O(1)}$

Worst Case

$\boxed{O(n)}$

Always look for the worst case.

Advance Problem

10:25

```
func(N, K) {
    for (i=1; i<=N; i++) {
        p = power(i, K)
        for (j=1; j<=p; j++) {
            print("yayyy")
        }
    }
}
```

i/p values

$p = i^K$

power() → O(1) =

Time complexity ?

i	j[1 i^K]	Iterations
1	[1 1^K]	1^K
2	[1 2^K]	+ 2^K
3	[1 3^K]	+ 3^K
4	[1 4^K]	+ 4^K
⋮		⋮
N	[1 N^K]	+ N^K

Iterations :

$1^K + 2^K + 3^K + 4^K + \dots N^K$

$K=1$: $1 + 2 + 3 + 4 + \dots N$ $\left\{ \frac{(N)(N+1)}{2} \right\} = \frac{N^2}{2}$ $O(N^2)$

$K=2$: $1^2 + 2^2 + 3^2 + 4^2 + \dots N^2$ $\left\{ \frac{(N)(N+1)(2N+1)}{6} \right\} = \frac{N^3}{3}$ $O(N^3)$

$K=3$: $1^3 + 2^3 + 3^3 + 4^3 + \dots N^3$ $\left\{ \left(\frac{N(N+1)}{2}\right)^2 \right\} = \frac{N^4}{4}$ $O(N^4)$

$\vdots$

K $1^K + 2^K + 3^K + \dots N^K$ $= \boxed{\frac{N^{K+1}}{K+1}}$

$O\left(\frac{N^{K+1}}{K+1}\right)$

$\boxed{O\left(\frac{N^{K+1}}{K+1}\right)}$

```
for (i = n; i > 0; i /= 2) {
    if (i % 2 == 0) {
        for (j = 1; j <= n^2; j += 2) {
            // O(1)
        }
    }
}
```

i	j	Iterations
n	$[1 \quad n^2]$	$(n^2+1)/2$
		+
$n/2$	$[1 \quad n^2]$	$(n^2+1)/2$
		+
$n/4$	$[1 \quad n^2]$	$(n^2+1)/2$
		+
$\vdots$		$\vdots$
1	$[1 \quad n^2]$	$(n^2+1)/2$

$\log_2 n * \frac{(n^2+1)}{2 \times 2}$

Next week. (2-3).

TLE [Time Limit Exceeded]

$$\boxed{\log_2 n * \frac{(n^2+1)}{4}}$$

$O(n^2 \log_2 n)$

Rajvardhan

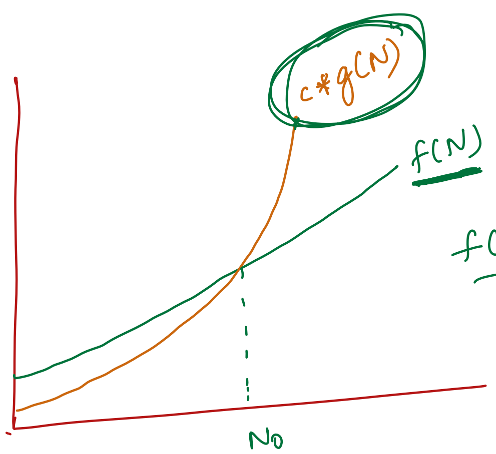
Google — 1, 2 → wrote the Approach → TLE → wrote the Approach → TLE

$O(1) < \log_2 N < \sqrt{N} < N < N\log N < N\sqrt{N} < N^2 < 2^N < N! < N^N$

logarithmic, linear, quadratic, exponential

All the iterations are func of N

f(N) = O(g(N)) if there exists two constants C & N_0 s.t. $C * g(N) >= f(N) \ \forall\ N >= N_0$.



f(N) = O(g(N))

$f(N) = O(g(N))$ if $f(N) <= c * g(N) \quad \forall N >= N_0$

$f(N) = N^2 + 5N \qquad g(N) = 2N^2$

N			
3	24	>	18
4	36	>	32
5	50	=	50
6	66	<	72
7	84	<	98
⋮		<	
⋮		<	

$$\boxed{N^2 + 5N <= 2N^2 \qquad \forall N >= 5}$$

$$f(N) <= c * g(N) \qquad \forall N >= N_0$$

$$g(N) = N^2$$

$$f(N) = O(N^2)$$

$$f(N) = \boxed{N^2 + 5N} \rightarrow O(N^2)$$