Vishank

15 sec

Windows 98

Mocbook

8 sec

(C++)

8 sec

7 sec

(Temp)

Shweta

10 sec

Mocbook

10 sec

(python)

7 sec

7 sec

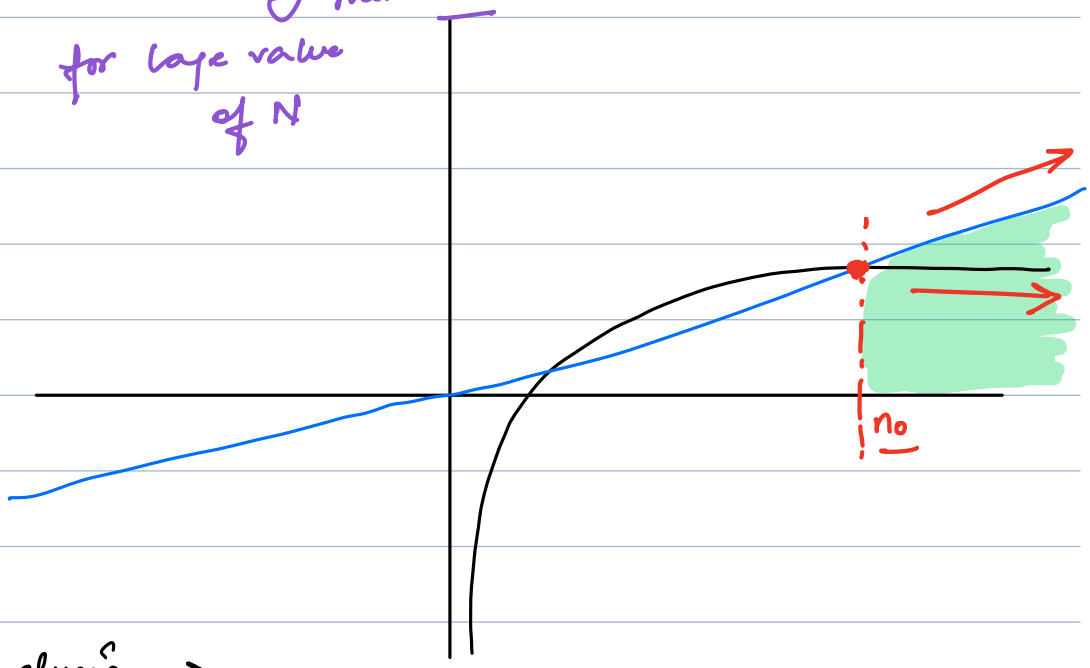
python is slower

Running time → f^n of input size

iterations

$10 \log_2 N < N/10$

for large value of N ↑ more true



Asymptotic analysis → when input size is large

↓

rate of growth

↓

Big O (notation)

↓

worst case?

↓

$4n^2 + 3n + 5$ ↑

↓

How to find Big-O:

1) Calculate no of iterations as a f^n of input size

2) Ignore lower order terms

3) Ignore constant coefficients

$$F(N) = 4N^2 + 3N + 1$$

$= O(N^2)$

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

$= O(N \log_2 N)$

$$1 < \log_2 N < \sqrt{N} < N < N\log_2 N < N\sqrt{N} < N^2 < N^2 \log_2 N < N^3 < 2^N < N! < N^N$$

logarithmic — linear — polynomial — exponential

Why lower order terms are ignored?

$N^2 + 10N$

$N = 10$	$(10)^2 + 10*10 = 200$	$100/200 = 50\%$
$N = 100$	$10^4 + 10^3 = 11000$	$\frac{10^3}{10^4+10^3} \approx 9\%$
$N = 10^5$	$10^{10} + 10^6$	$\frac{10^6}{10^{10}+10^6} \approx 0.01\%$

$N \log_2 N$ → can we ignore

↓

depends on input

loopholes → ① for larger inputs only, never takes into account smaller inputs

② $10N^2 + 5N + 6$ → $O(N^2)$ $\quad$ $7N^2 + 4N + 3$ → $O(N^2)$

③ constant coefficients are too large to ignore

```
for( int i=1; i<=100; i++)
{

}
```

→ 100 → $O(1)$

$N = 10^6$

$19n$ → $19 * 10^6$

$N \log_2 N$ → $10^6 * 19$

- search for an element k in the array

↓

linear search

```
for( i=0; i<N; i++)
    if(arr[i]==k)
        return true;
```

Best case → 1

Worst case → N

Q sum of elements in the array

```
sum=0;
for( i=1; i<N; i++)        → N iterations
    sum+=arr[i];
```

Space complexity → extra space in terms of input

```
fume ( int N )          ← 4B
{
    int a;              ← 4B
    int b;              ← 4B
    long c;             ← 8B
}
```

= 20 Bytes

↓

space is not growing with input

↓

O(1)

```
void fun ( int N ) {        ← 4B
    int arr[10];            → 10×4 = 40 B
    float f;                → 4B
    int arr2[N];            → 4N
}
```

4N + 48

∴ O(N)

```
void fun ( int N ) {        ← 4B
    int arr[N];             → 4N
    float f;                → 4B
    int arr2[N][N];         → 4N²
}
```

4N² + 4N + 8 → O(N²)

↓ Time complexity ⟷ ↓ space complexity

Time limit exceeded (TLE)

↓

1 sec – 2 sec 1 GHz = 10^9 cc/sec

```
int sum = 0;
for (int i=0; i<N; i++)
    sum = sum + arr[i];
return sum;
```

≈ 10 operations

1 sec = 10^9 oper

1 iter ≈ 10 oper

≈ 10^8 iterations

1 sec ≈ 10^6 – 10^7 iterations

constraints

$1 \le N \le 10^5$

$O(N^3) \rightarrow 10^5 * 10^5 * 10^5 = 10^{15}$

$O(N^2 \log_2 N) \quad 10^5 * 10^5 * 10 = 10^{11}$

$O(N^2) \quad 10^5 * 10^5 = 10^{10}$

$O(N \log_2 N) \quad 10^5 * 10 = 10^6$

$1 \le N \le 10^6$

N^3 ✗

$N^2 \log_2 N$ ✗

N^2 ✗

$N \log_2 N = 10^6 * 10 \approx 10^7$

$O(N)$ ✓

MLE → memory limit exceeded

$1 \le N \le 100$ (10^2)

$O(N^3)$

2^N

$1 \le N \le 16$ or $1 \le N \le 20$

$n!$ $1 \le n \le 10$

$1 \leq N \leq 10^9$

$O(1)$ $O(\log_2 N)$