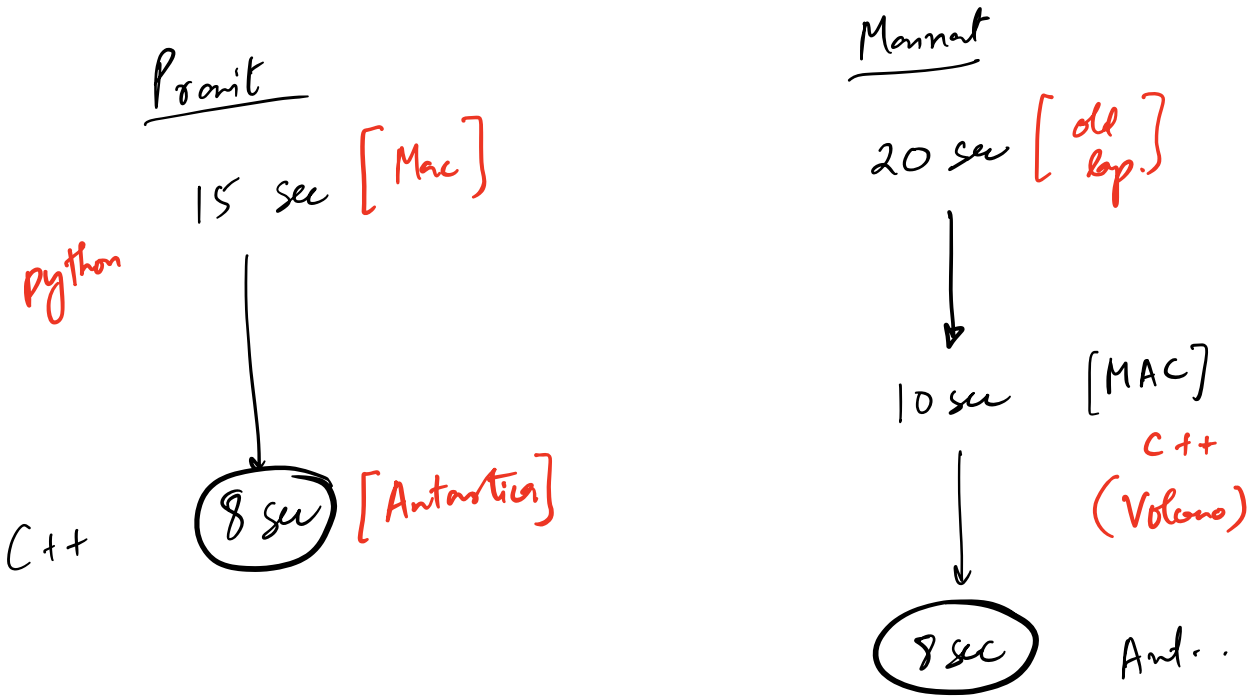
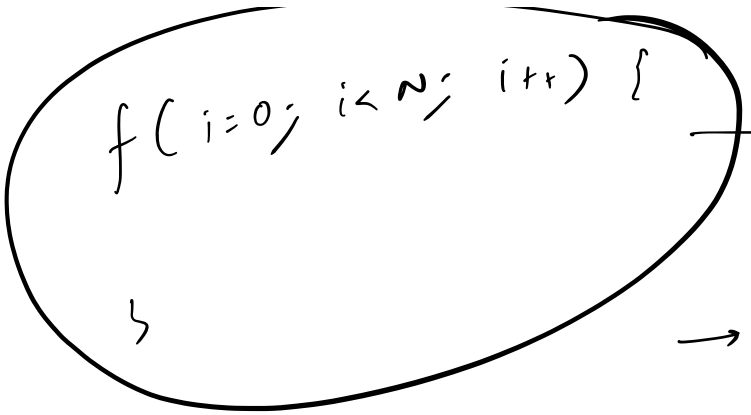
Time Complexity - 2

1) Given 10^5 ints. Sort them in ASC order!



Execution time is not a good factor to judge Algo's

↳ SW + HW + External cond.-s



→ #ops / iter ... → N
do not change with any factor.
→ they remain constant

#

SORT → N elements

Profit VS Manual

#ops → $100 \log_2 N$

$N/10$

N = 32

$100 \log_2 32$

100×5

$= 500$ ops

$32/10$

~ 3 ops ✓

N = 64

$100 \times \log_2 64$

100×6

$= 600$ ops

$64/10$

~ 6 ops ✓

$N = 2^{20}$

$100 \log_2 2^{20}$

100×20

2000 ops ✓

$2^{20}/10$

$\sim 10^6/10$

$\sim 10^5$

$N = 2^{40}$

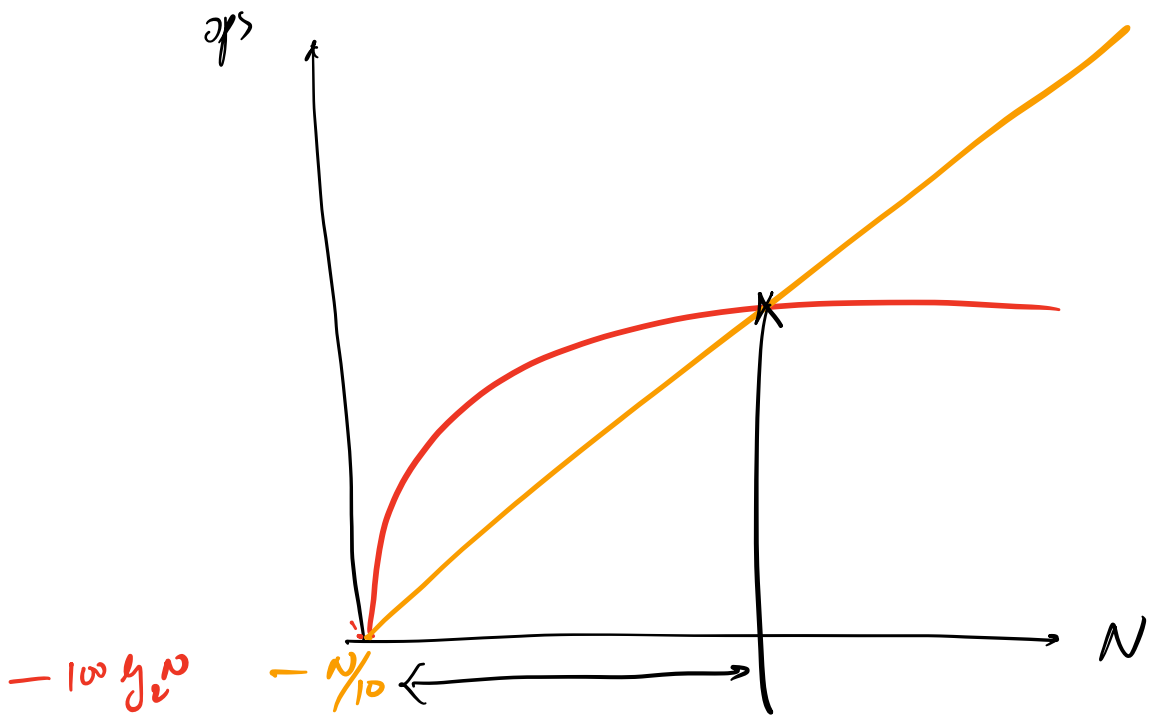
100×40

$= 4000$ ops ✓

$10^{12}/10$

$\sim 10^{11}$

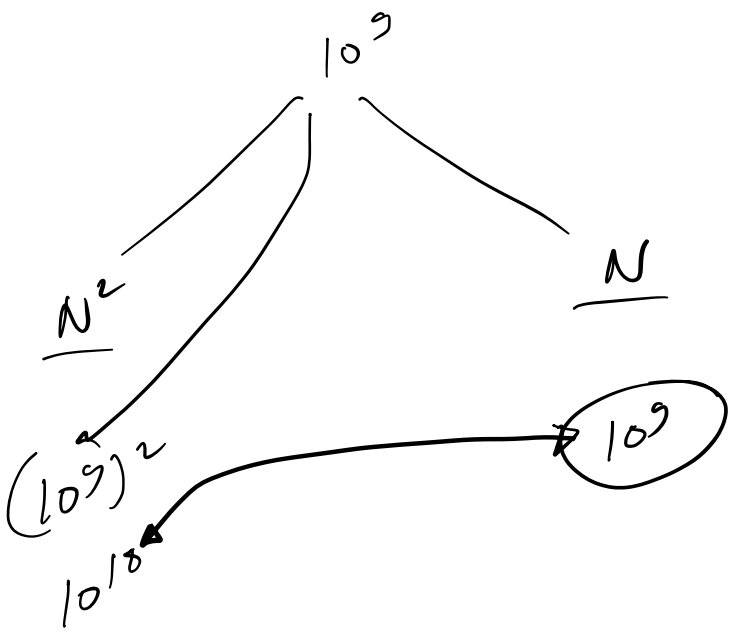
$(100 \log_2 N)$ would perform better for larger values of N. compared to $(N/10)$



⊙ Algo 1 Algo 2 Algo 3 ---

Hotstar	Youtube	Google Search
5×10^6	2×10^9	$\sim 10^{10}$

Input values are generally large for Applications.



⊙ Asymptotic Analysis of Algo's

↳ Used to judge the performance of an algorithm for large input sizes!

$N = 10^8$

N

10^8 ops

1 sec

$N\sqrt{N}$

$10^8 \times \sqrt{10^8}$

$10^8 \times 10^4$

$= 10^{12}$ ops

N^2

$(10^8)^2$

$= 10^{16}$ ops

$10^{16} \rightarrow \frac{10^{16}}{10^8}$ sec

$\rightarrow 10^8$ sec

$\sim 30{,}000$ hrs

10^8 ops $\rightarrow$ 1 sec

1 op $\rightarrow \frac{1}{10^8}$ sec

10^{12} ops $= 10^{12} \times \frac{1}{10^8}$ sec

$= 10^4$ sec

~ 3 hrs

Big O Notation

1) Calculate the no. of ops w.r.t input
2) Neglect lower order terms.
3) Neglect the constant coefficient!

```
cnt = 0, x = 0
for( i = 0; i < N; i++) {
    cnt++;
    x--;
}
```

- #it → N
- #ops → 4N
- #INST → X

10^8 ops → 1sec

1GHz = 10^9 clock cycles per sec

Amit →

N → Input

$N^2 + N$ ops

order: 2 1

↓

N^2

TC: $O(N^2)$

N

ops | $f(N) = 10N^2 + 50N + 600 \cdot N^0$

$10N^2$

TC : $O(N^2)$

Input → N

ops → $f(N) = 600N^3 + 5N^2 + 5N\sqrt{N} + 46 \lg N$

$600N^3$

TC : $O(N^3)$

⊙ I/P → N

$f(N) = 60N^3 \lg N + 45N^3$

TC : $O(N^3 \lg N)$

⊙ I/P → N, M

$f(N,M) = 40N^3 + 20N^2 + 59M\sqrt{M} + 10M + 60$

$40N^3 + 59M\sqrt{M}$

$TC = O(N^3 + M\sqrt{M})$

I/P → N

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

% contribution of N

$= \frac{100}{10100} \times 100\%$

$= \sim 1\%$

$N = 100$

$f(100) = (100)^2 + 100$

$= 10000 + 100$

$= 10100$

$N = 10^5$

$f(10^5) = (10^5)^2 + 10^5$

$= 10^{10} + 10^5$

$\frac{10^5}{10^{10} + 10^5} \times 100\%$

$\sim 0.001\%$

$f(n) = 50\ 2^n + 45\ n^2 \sim$

$TC = O(2^n)$

$2^{10} = 1024 \simeq 10^3$

$2^{20} \simeq 10^6$

$2^{30} \simeq 10^9$

$2^{60} \simeq 10^{18}$

$10^3 \times 10^2 \rightarrow 10^5$

$2^{10} \times 2^6$

$\sim 2^{16}$

```
for(i=0 — N-1){
    cnt++;
}
```

→ # it → N

ops → 2N

TC : O(N)

```
for( i=0 → N-1) {
    S+="          "
}
```

M

~ M ops

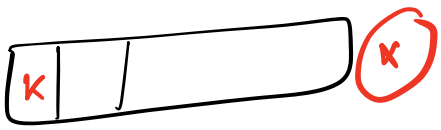
it → N

ops → N

TC : O(NM)

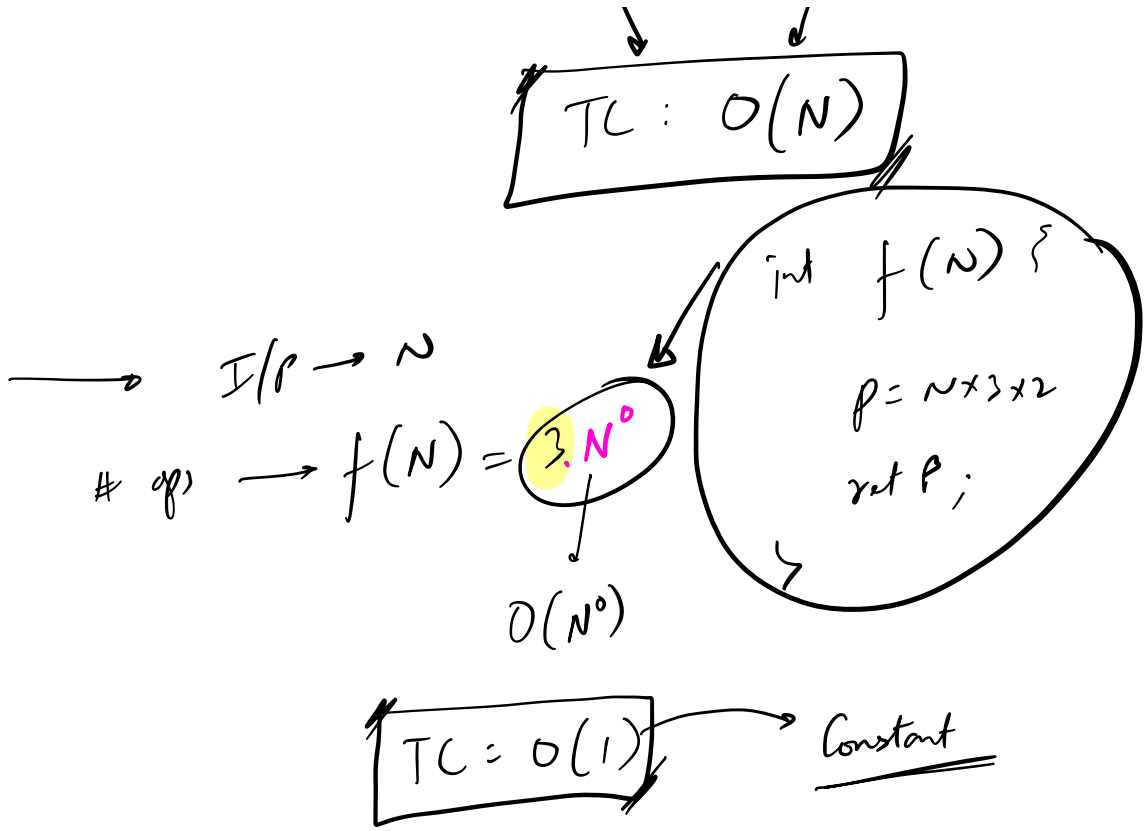
Q Given an array, find K in it.

```
for(i=0; i<N; i++){
    if( A[i] == K){
        ret True;
    }
}
ret False;
```



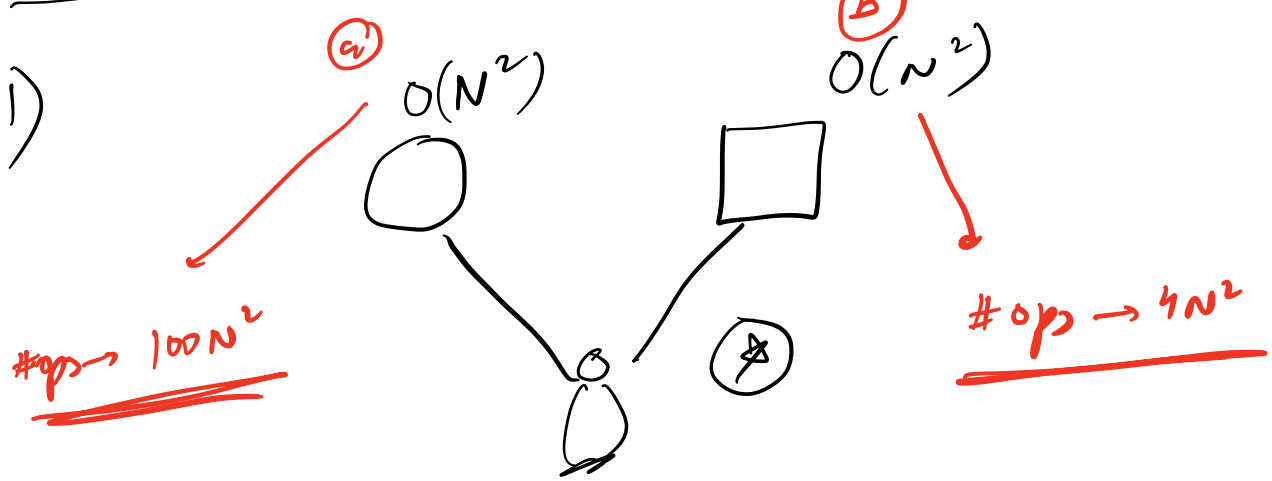
→ Best Case → # 1 op

Worst Case → # N ops



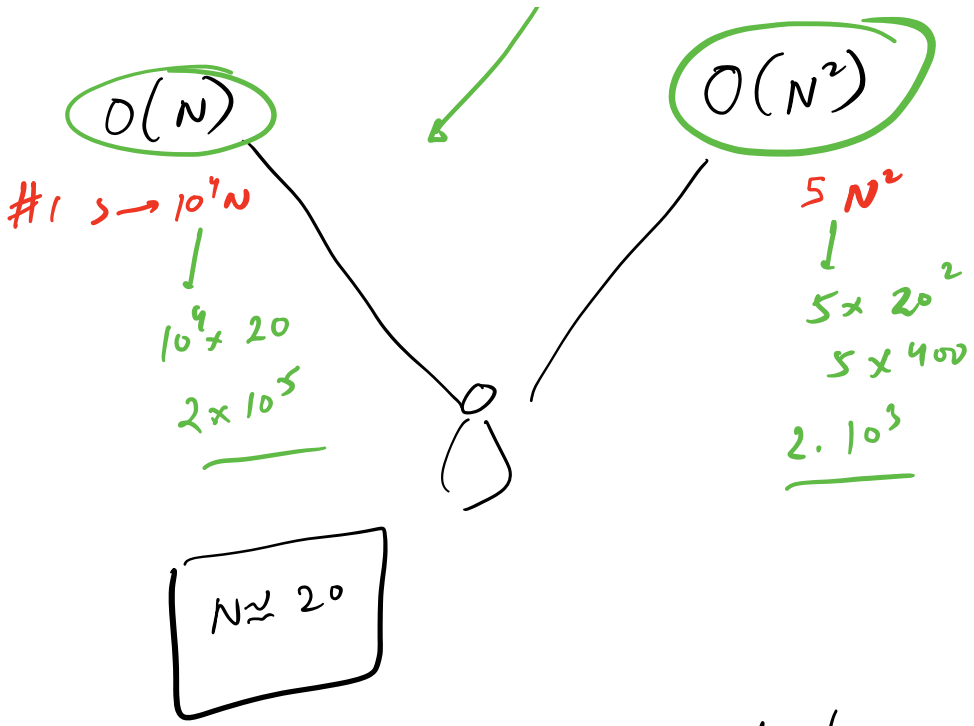
Issues with Big O

1)



We cannot get an exact idea of #ops

2)



→ We can't decide which algo is better for smaller values of Input!

Space Complexity → Big O

int → 4B
double → 8B
long → 8B

```
func(N) {
    int x = N;        → 4B
    int y = 10;       → 4B
    double z = 20;    → 8B
    long v = 400;     → 8B
}
```

24 B

#bytes → $f(N) = 24$

$= 24 N^0$

$O(N^0)$

$SC = O(1)$ Constant

```
func(N) {
    int x = N;        → 4B
    int y = 10;       → 4B
    long z = 50;      → 8B
    int A[N];         → 4N B
}
```

#bytes → $f(N) = 16 + 4N$

$SC = O(N)$

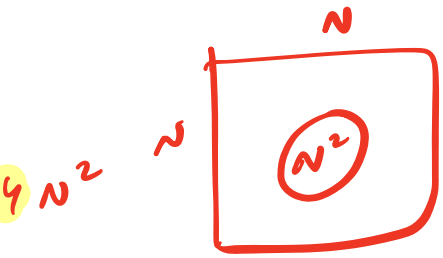
N

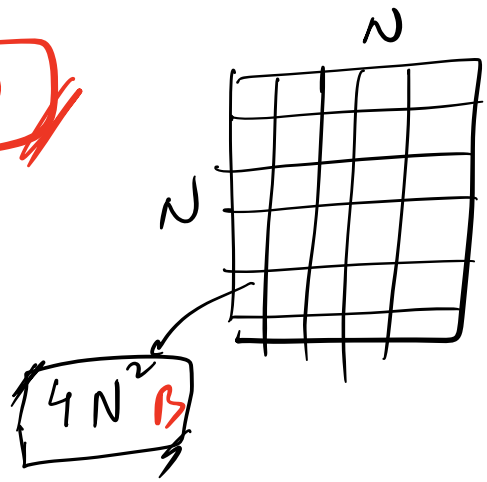
int
4B 4B

```
func ( N)
    int x = 10;   → 4B
    int y = N;    → 4B
    int A [N];    → 4N B
    int B [N][N]; → 4.N² B
}
```

$f(N) = 8 + 4N + 4N^2$

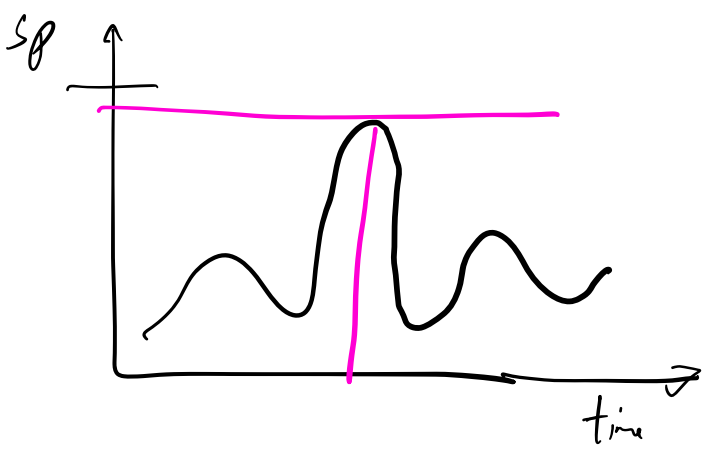
$SC = O(N^2)$





Space Complexity:

MAX amount of space your algo is taking at any point in time



⊙ TLE → Time Limit Exceeded !

ⓐ N

ops → N^2

ops → $(10^5)^2 = 10^{10}$

TL : 1sec / 2sec
ML : 256 MB

$1 <= N <= 10^5$
$1 <= A_i <= 10^9$

10^8 ops → 1sec

10^{10} ops → 10^2 sec → TLE !

ⓑ # ops → 50N

50×10^5

$= 5 \times 10^6$ → < 1 sec

Prob → Algo → Calc # ops

→ calc the time taken → $\tau = 2$ sec ?

↳ XILE