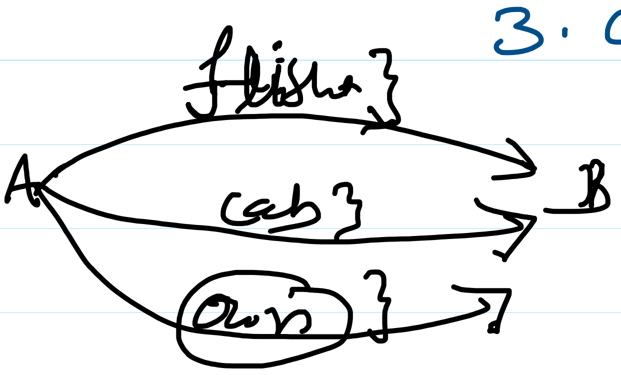
→ Algorithms

1. think / read
2. Devise the algo
3. code



Cost.

Q: Goal of algo. anal.

1. Time
2. Space complexity.

A B

1. leetcode
2. Hackerrank }

Relevel

↳ Soln
↳ Time
↳ Space.

{ Alpha } { Same }
{ word }

Q: How to do Algo. Anal.

1. Experimental analysis:.

{5}

10000

Exam.

Aradhya — mac → } 3 secs.

Nishant — Pen-paper } 1 hr.

{ Machine dependent }

{ Comparable }

2. Asymptotic analysis.

↳ Compare algo = { input growth

Aradhya

Nishant

Sol^n 1 — 3sec

n=4 ≈ 4

4.5 sec

n=100 ≈ 100

3sec — Sol^n 2

n=4

1 min

n=100

n=4

1, 2, 3, 4 } 4 steps

- Iterate through array
- Print no.

1. Best ⇒ Ω( )
2. Avg. ⇒ Θ( )
3. Worst ⇒ O(_)

MAP

[ ] → 10 hrs

MAP 50 hrs

← 50 hrs + 10 hrs

Worst — O(n)

Best & Avg <

I/P- [0, 1, 0, 0, 0, 1] ⇒ O/P [0, 0, 0, 0, 1, 1]

[1, 1, 0, 0, 0, 0] — Print all 0's

I/P [0, 0, 0, 0, 1, 1] — Best Case

} Best — 0 ns, 1 lac, 1 cn, 1000 cn

[1, 1, 1, 1, 1, 0] Worst } 5 sec

[0,1,1,1,1,1,1]

Best
Avg } < 5 sec

i

y = [50 + (100 + 20) * 6]

4 steps

O(4) → Completes steps

O(1)

Q

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

O(n)

n=3
i=0
i=1
i=2
} 3 times

```
for(j=0; j<n/2; j++)
{
    print(j)
}
```

n=6
j=0   3
j=1   1
j=2   0

O(n/2)

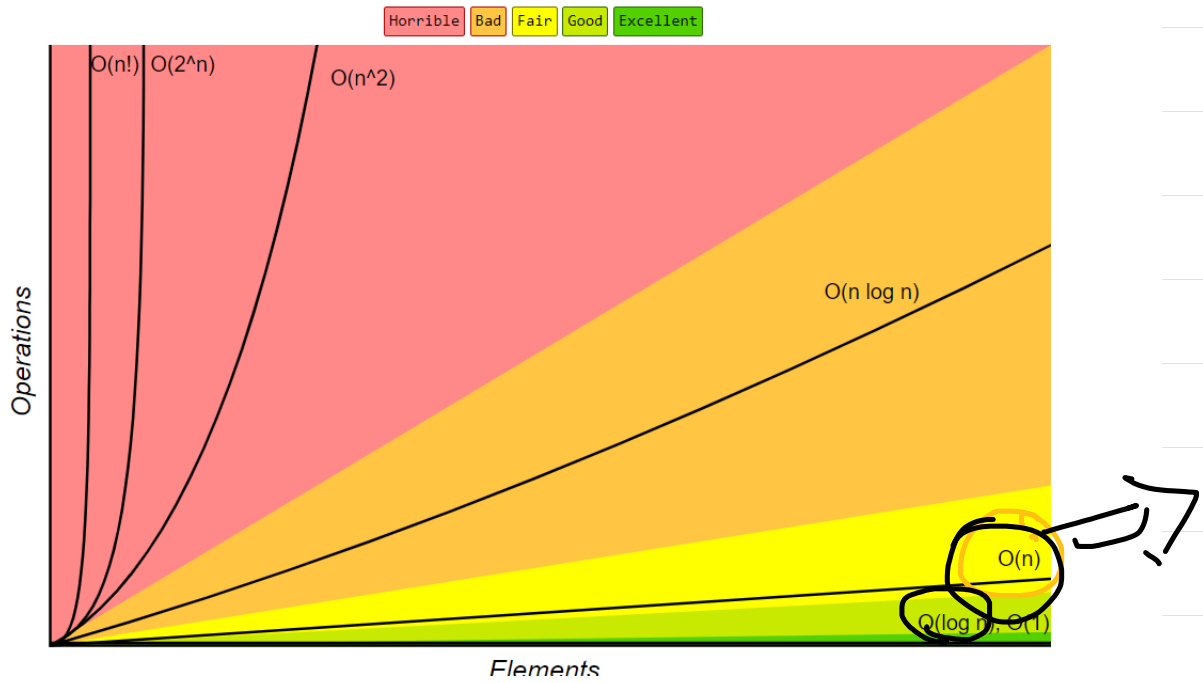
Sequential => Add O(n) + O(n/2)

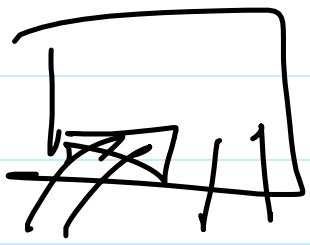
O(n) + O(log n)

O(n) + O(1/2 n)

O(n) + O(n)

O(n)





n = 1m.

10000000000 + 100

O(1) ⇒ constant

O(n) ⇒ Linear

$O(n^2)$ ⇒ quadratic

O(log n) ⇒ logarithmic

O(n!) ⇒ factorial

n^n

n * m

n = m

for(i=0; i<n; i++). O(n)

n < m