# Binary Search

→ Optimized Search Algo

→ Uses divide and Conquer strategy

→ Search space MUST be SORTED

[1, 2, 3, 4, 10, 20, 35] → Ascending

[25, 16, 12, 10, 5, 3, -10, -20]
Descending

## Algorithm:

1) Find the middle Element
2) target > mid ⇒ Search in the right
   else search in left
3) if middle == target element // ans

Eg

arr = [1, 4, 5, 7, 8, 10, 35, 40]
(0 1 2 3 4 5 6 7)
s = 0, e = 7

Target = 8

$m = \frac{s+e}{2} = \frac{0+7}{2} = 3.5 \approx 3$

arr = [1, 4, 5, 7, 8, 10, 35, 40]
s = 0, m = 3, e = 7

?? 7 >=< 8
⇒ 7 < 8

Therefore Look on the Right Side

⇒ s = m + 1
= 3 + 1 = 4

arr = [1, 4, 5, 7, 8, 10, 35, 40]
s = 4, e = 7

$m = \frac{s+e}{2} = \frac{4+7}{2} = \frac{11}{2} = 5.5 \approx 5$

arr = [1, 4, 5, 7, 8, 10, 35, 40]
s = 4, m = 5, e = 7

?? 10 >=< 8

10 > 8 ⇒ Look on Left side of m

∴ e = m − 1
= 5 − 1 = 4

arr = [1, 4, 5, 7, 8, 10, 35, 40]

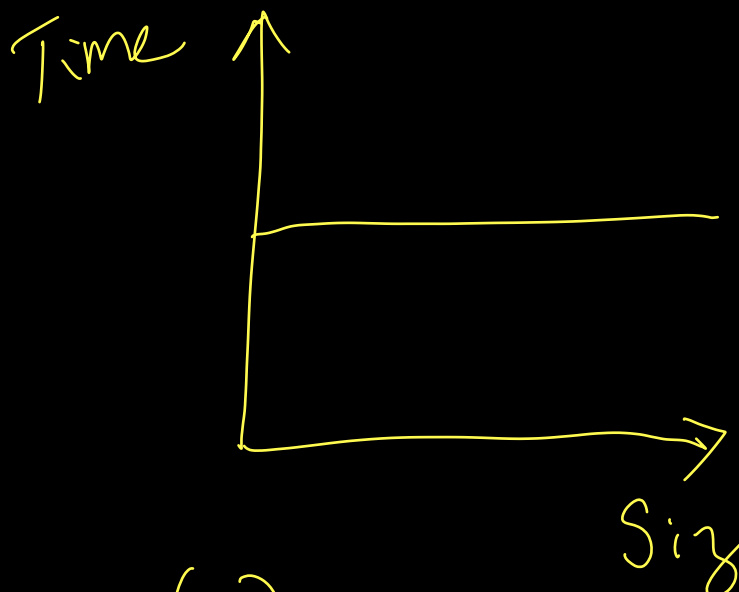
0 1 2 3 4 5 6 7

s e m

$m = \frac{s+e}{2} = \frac{4+4}{2} = 4$

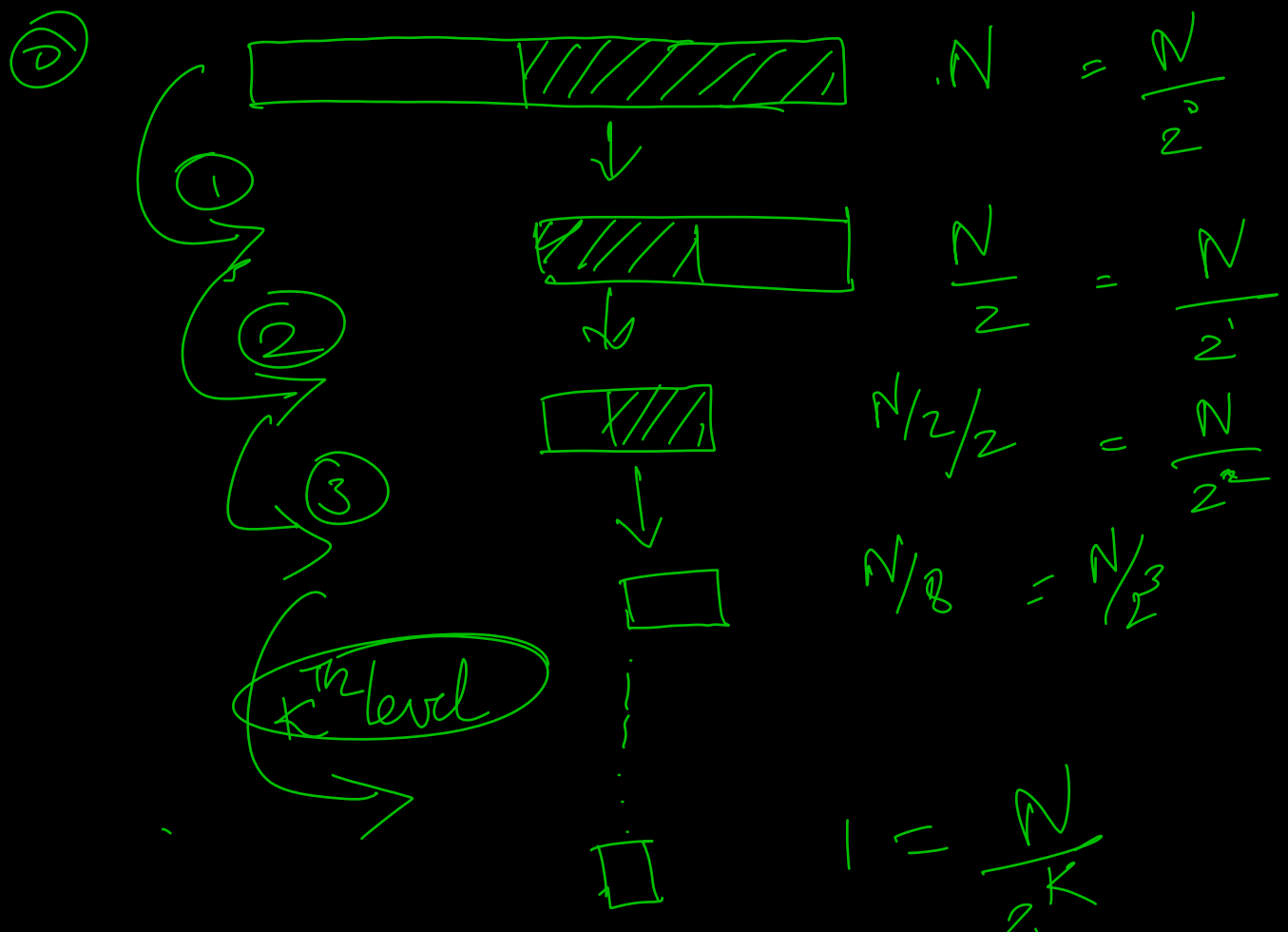
" 8 is found at Index 4 ".

if s > e : element not found




O(1) → Best case (Target is mid elemt

Q. Find the max # of comparisions in worst case?




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

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

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

$$K = \frac{\log N}{\log 2} = \log_2 N$$

$\therefore$ Worst case $= O(\log N)$

Compare for $N = 1,000,000$

| Linear | Binary |
| --- | --- |
| 1 million comparisions | $\log_2 1000000$ = 20 comparisions |

// better way to find mid

* $m = \frac{s+e}{2}$ → This may exceed the int range

$$m = s + \frac{(e-s)}{2}$$

$$s + \frac{(e-s)}{2}$$

$$\frac{2s + e - s}{2}$$

$$= \frac{s+e}{2}$$