arr = [2, 4, 9, 10, 12, 14, 18, 19]

→ ascending

arr2 = [19, 12, 6, 5, 3, 2, −8, −16]

→ descending order

max comparisons : N ⇒ No. of elements

+9 = 4

① find the middle element

② target > mid ⟹ search in the right
else search in left

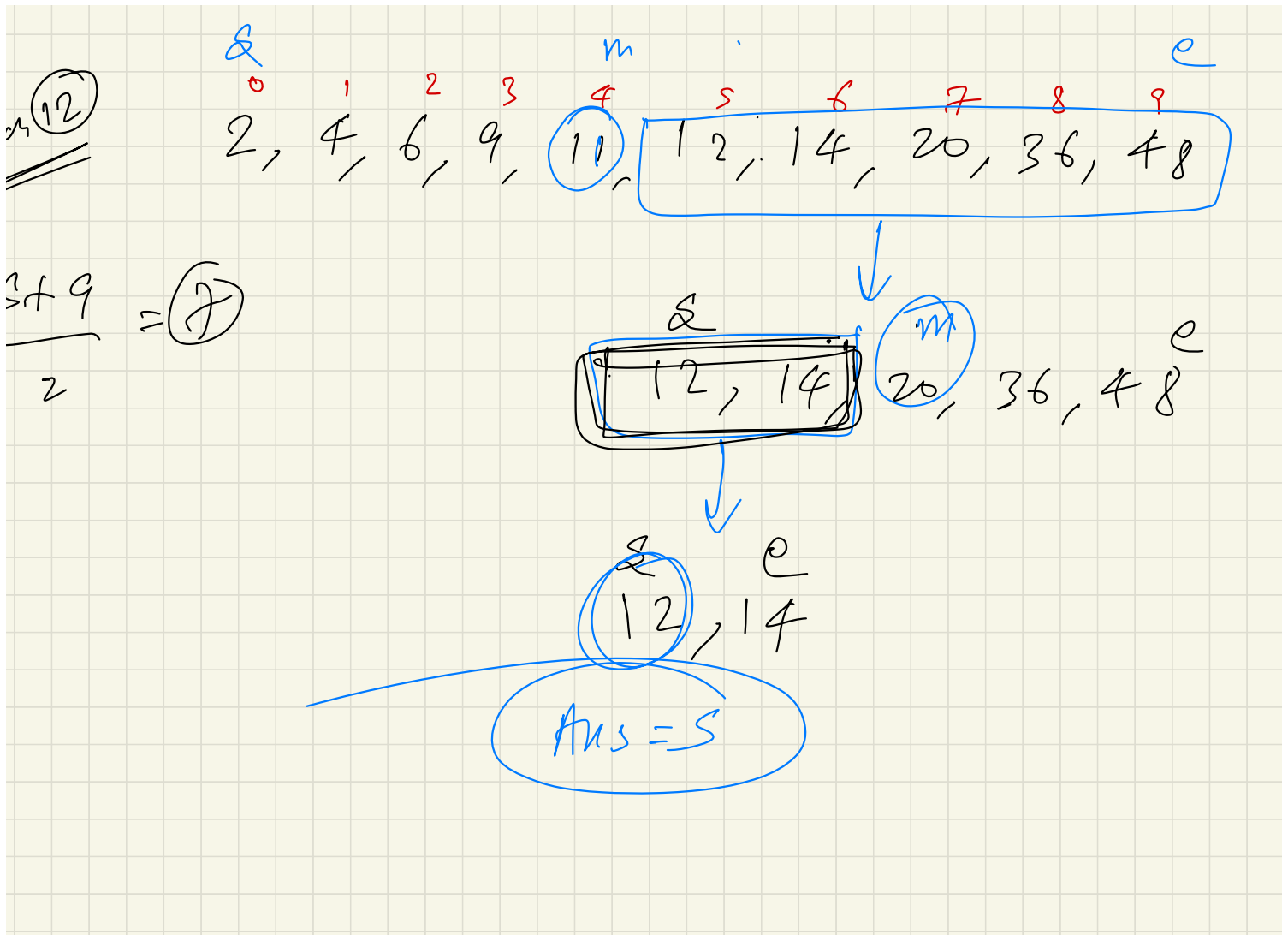
③ if middle element == target element //ans

s m e

0	1	2	3	4	5	6	7	8	9
2	4	6	9	11	12	14	20	36	48

arr = [2, 4, 6, 9, 11, 12, 14, 20, 36, 48]

target = 36


s
m
e
0 1 2 3 4 5 6 7 8 9
2, 4, 6, 9, 11, 12, 14, 20, 36, 48
12
s+9 / 2 = 7
s
m
e
12, 14
20, 36, 48
s
e
12, 14
Ans = 5

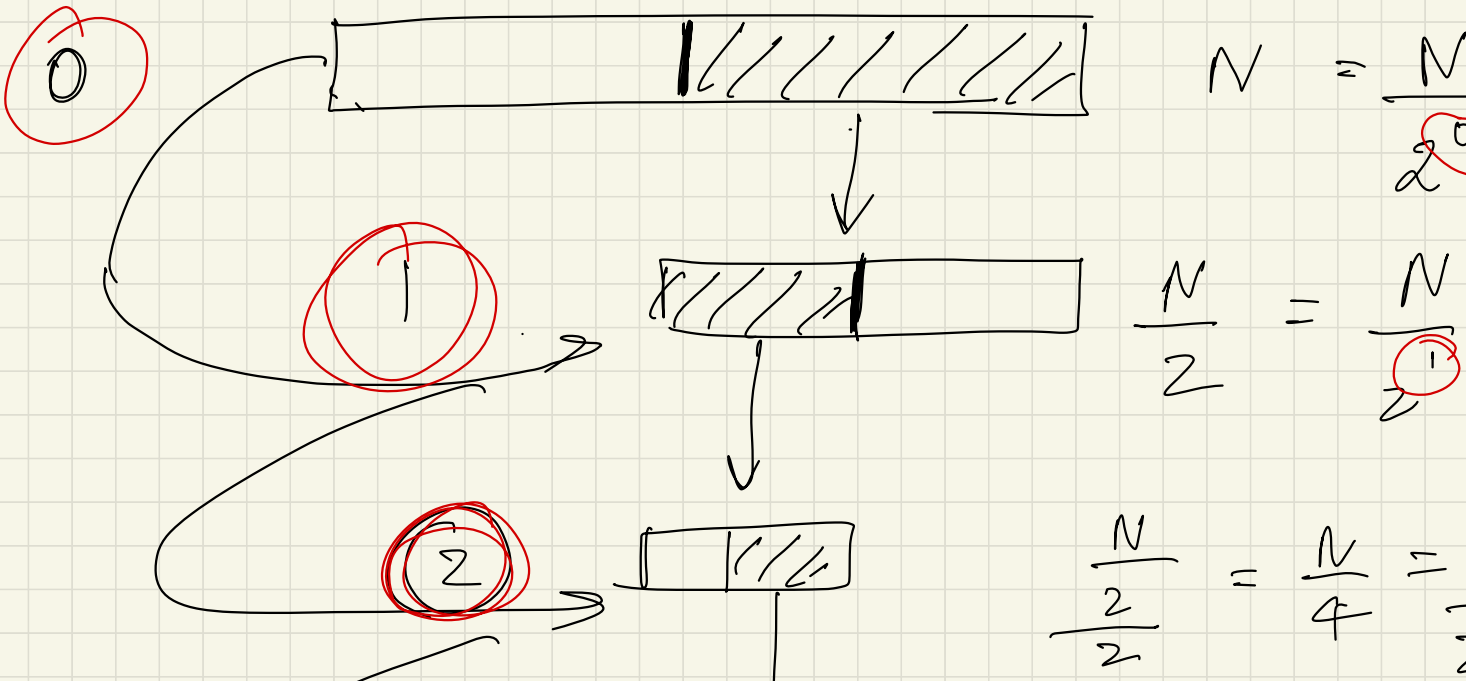
if $s > e$: element not found.

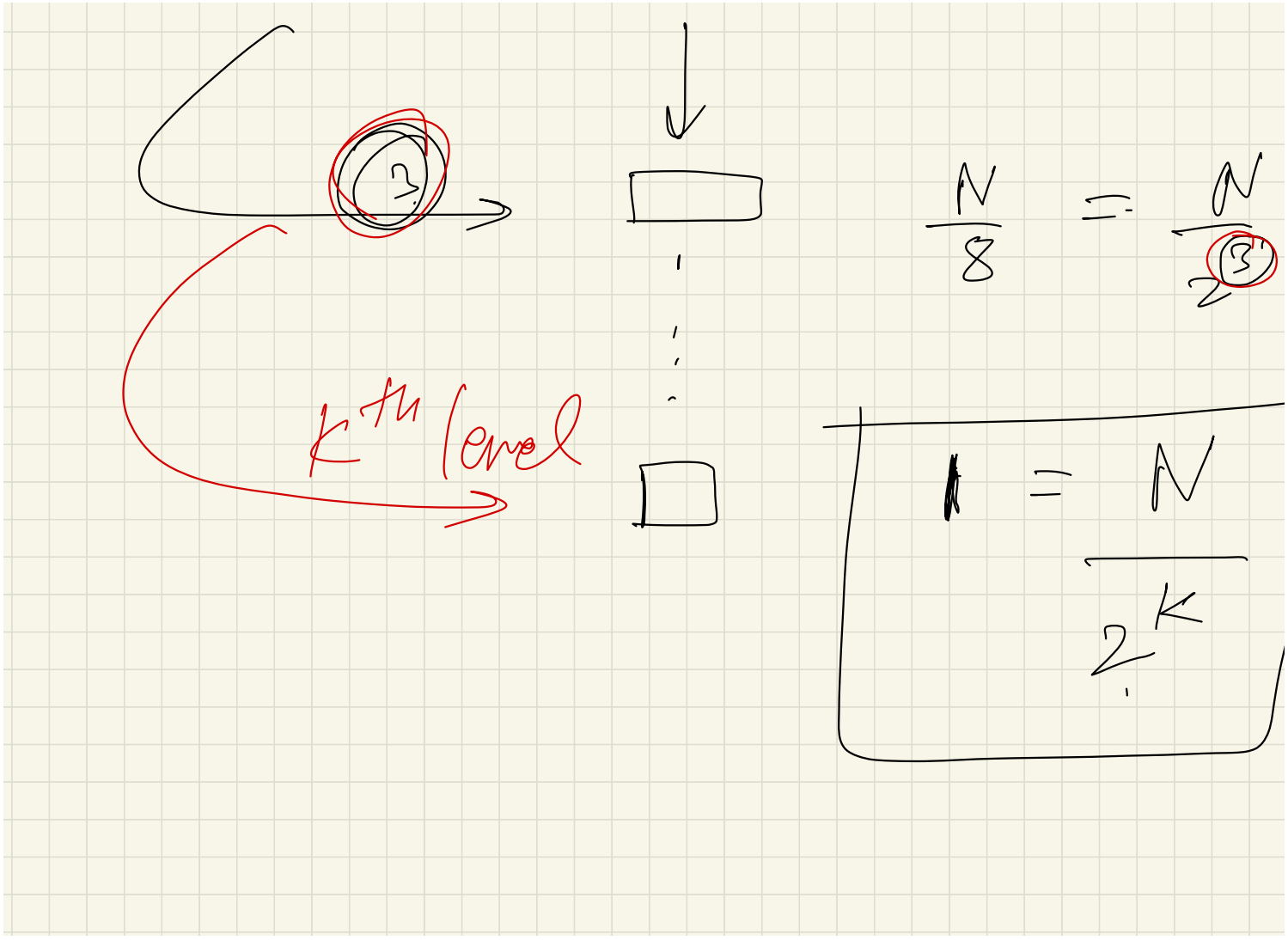
Best Case

$O(1)$

Why Binary Search?

Q: Find the max number of such comparisons in worst case.




1
k^{th} level
$\frac{N}{8} = \frac{N}{2^3}$
$1 = \frac{N}{2^k}$

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

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

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

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

Total comparisons in the worst case = log

Search in a 1,000,000

Linear

1 mill

Binary search

20 comparisons

O(logN)

// better way to find mid

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

✡ $m = s + \frac{(e-s)}{2}$

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

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

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

#dsawithKunal

@commclassroom

@Kunalstwt

Order agnostic Binary Search

arr = [90, 75, 18, 12, 6, 4, 3, 1]

s = 0, m = 3, e = 7

target = 75

target > middle ⇒ left

e = m - 1

target < middle ⇒ right

s = m + 1

arr = [3, 3, 3, 5, 8, 9, 12, 14, 20, 33]

i j Prob!

if s > f ⇒ increasing

else ⇒ decreasing