BINARY SEARCH ALGORITHM

arr = [2 , 4 , 6 , 9 , 11 , 12 , 14 , 20 , 36 , 48]
0 1 2 3 4 5 6 7 8 9

target = 36

ALGORITHM :-

① Step 1 :- Find the middle element.

② Step 2 :- Check :-
if target > middle ⇒ search in right
else ⇒ search in left.
if target == middle ⇒ we found element.

In this example :-

① middle = $\frac{start + end}{2} = \frac{0+9}{2} = 4.5 = (4) \Rightarrow$ i.e 11.

② is 36 > 11 ⇒ yes ⇒ check in right side.

③ Now arr = [2, 4, 6, 9, 11, 12, 14, 20, 36, 48]
S = 0, m = 4, e = 9; 5 6 7 8

④ Now S = 5 e = 9, mid = $\frac{5+9}{2} = 7$

⑤ 36 > 20 ⇒ yes ⇒ check in right side.

⑥ arr = [2, 4, 6, 9, 11, 12, 14, 20, 36, 48]
0 1 2 3 4 5 6 7 8 9; s = 8, e = 9

⑧ m = $\frac{8+9}{2} = 8$

⑨ 36 = 36 ⇒ ELEMENT FOUND AT INDEX 8

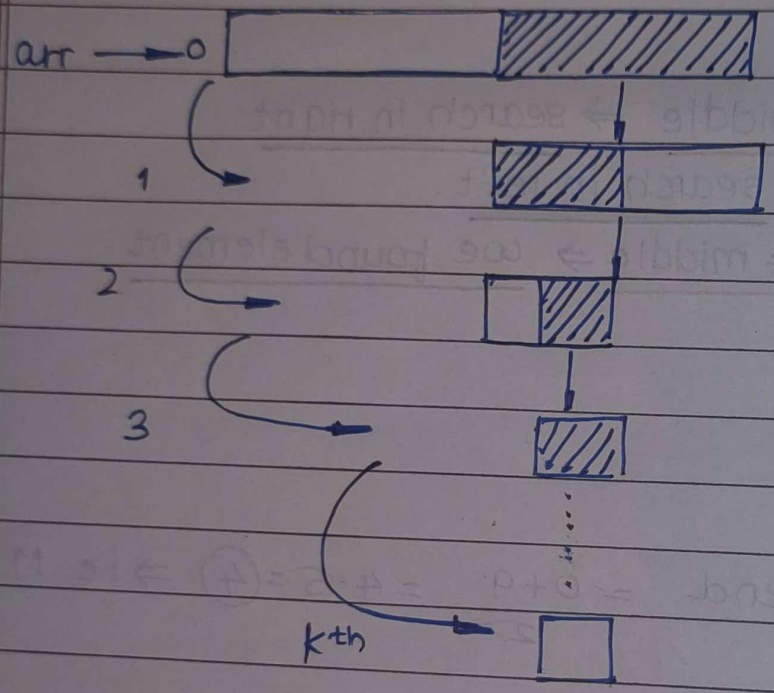
TIME COMPLEXITY.

Best case:- O(1)

⇒ element is present at the middle.

Worst case:- O(log n)

Explanation:- Max. number of comparisons.



$N = N/2^0$

$\frac{N}{2} = N/2^1$

$\frac{N}{4} = N/2^2$

$\frac{N}{8} = N/2^3$

$(1) = \frac{N}{2^k}$

$\therefore \frac{N}{2^k} = 1 \Rightarrow N = 2^k$

$\log N = \log 2^k$

$\log N = k \log 2$

$\boxed{k = \log_2 N}$

k → total no. of comparison in worst case

N → size of array.

* Order Agnostic Binary Search.

```
          s                 m                 e
arr = [90, 75, 18, 12, 6, 4, 3, 1]
       0   1   2   3   4  5  6  7
```

target = 75

target > middle ⇒ left ⇒ e = mid - 1
target < middle ⇒ right ⇒ s = mid + 1.

Here, start > end] —— Deseending order

start < end] —— Ascending order.