BINARY SEARCH

Binary Search is an algorithm for finding an element in a sorted array. it repeatedly divides the search interval in half, ~~reducing~~ comparing middle element to target value & then searching of either left or right half depending on comparision.

Why Binary Search :-

★ Binary Search is chosen over linear search ~~methoe~~ when data is shorted, because it offers much faster performance, high Efficiency & uses low memory.

★ When not shorted then Binary don't work.

Algorithm :-

Steps :- 0.> array should be shorted in ascending order.

1.> Find the middle element.

2.> Checks :-

```
    if target > middle => search in right
       else => search in left
```

3.> if target == middle → found the element.

array is sorted in ascending order.

Example 1 –

$$arr = [\underset{s}{\overset{0}{2}}, \overset{1}{4}, \overset{2}{6}, \overset{3}{9}, \underset{m}{\overset{4}{11}}, \overset{5}{12}, \overset{6}{14}, \overset{7}{20}, \overset{8}{36}, \underset{e}{\overset{9}{48}}]$$

target = 36

Step 1. Middle Element

$$mid = \frac{start + end}{2} = \frac{0+9}{2} = 4$$

Step 2. check

target > middle

=> 36 > 11 => check on right

Step 3 New array =>

$$[\underset{s}{\overset{5}{12}}, \overset{6}{14}, \underset{m.}{\overset{7}{20}}, \overset{8}{36}, \underset{e}{\overset{9}{48}}]$$

★ we are not making copy ~~just~~ of array. just ~~the~~ index for 0 to 4 are not used so no displayed in notes

$$Mid \Rightarrow \frac{5+9}{2} = 7$$

Check => targen > middle

36 > 20 => check on right.

New array ->

$$[\underset{\underset{(m)}{s}}{\overset{8}{36}}, \underset{e}{\overset{9}{48}}]$$

$$middle \Rightarrow \frac{8+9}{2} \Rightarrow 8$$ { here middle start comes same }

check => target = middle

36 = 36

Hence; element is fount at index 8.

Example 2:- target = 12

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

0 1 2 3 4 5 6 7 8 9

s m e

Step

1 middle = $\frac{s+e}{2} = \frac{0+9}{2} = 4$

2. check = target > middle ⇒ 12 > 11 ⇒ yes!

⇒ check on righ side.

Now arr = [12, 14, 20, 36, 48]

5 6 7 8 9

s m e

middle = $\frac{5+9}{2} = 7$

check = target < middle ⇒ 12 < 20 ⇒ check of left side.

5 6

Now arr = [12, 14]

s e

m

m

middle = $\frac{5+6}{2} = 5$

check = target = middle

⇒ 12 = 12

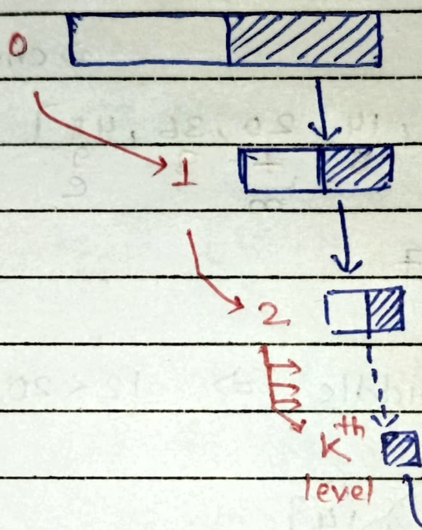
⇒ found at index 5.

Time Complexity :-

Best Case : O(1)

Worst Case : O(log n)

Explanation of maximum number of comparision:-



$N = N/2^0$

$N/2 = N/2^1$

$N/4 = N/2^2$

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

only one element left

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

$\Rightarrow \log N = k \log 2$

$\Rightarrow k = \frac{\log N}{\log 2.}$

$\therefore \log_2 N = k$

where :-

N = size of array

k = total number of comparision in worst case, etc.

Order Agnostic Binary Search :-

Order Agnastic Binary Search is a variation of binary Search algorithms that works on both ascending & descending sorted arrays.

Conditions :-

- Start > end ⟶ Descending order
- Start < end → Ascending order.

Example :-

$$arr = [\overset{0}{\underset{s}{90}}, \overset{1}{75}, \overset{2}{18}, \overset{3}{\underset{m}{12}}, \overset{4}{6}, \overset{5}{4}, \overset{6}{3}, \overset{7}{\underset{e}{1}}]$$

Here = s > e ⇒ 90 > 1 ⇒ descending order.

Now target = 75

- mid = $\frac{0+7}{2}$ = 3 & • target > middle
  ⇒ 75 > 12 ⇒ search left.

- new array = 90, 75, 18

now, mid = $\frac{0+2}{2}$ = 1

- target = middle
  75 = 75
  ⇒ found element / traget.