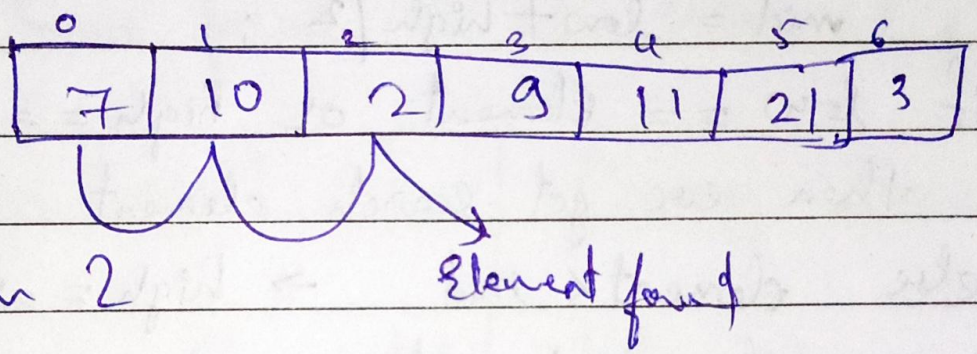
Linear & Binary Search

→ Linear Search :-

This search method searches for an element by visiting all the elements sequentially until the element is found or the array finishes.



→ Binary Search :-

This search method searches for an element by breaking the search space into half each time it finds the wrong element.
The search continues towards either side of the mid, based on whether the element to be searched is lesser or greater than the mid element of the current search space.

Search 2 in arr

0	1	2	3	4	5	6	7	8	9
1	2	8	15	19	23	29	48	73	79

low ↑ (index 0) high ↑ (index 9)

let low = 1 {arr[0]}
high = 79 {arr[9]}

→ if 2 (element) < high then continues or stop.
→ mid = low + high / 2 ;
→ if low == element or high == element or mide==de
then we got search element
→ else element < mid → high = arr[mid]
element > mid → low = arr[mid + 1]
After this again continue this step for these short arrays. Until low >= high

Search 2

0	1	2	3	4	5	6	7	8	9
1	2	8	15	19	23	29	48	73	79

low ↑ (0) mid ↑ (4) high ↑ (9)

mid = $\frac{0+9}{2}$ = 4.5 ≈ 4

1 ≠ 2 or 79 ≠ 2 or 19 ≠ 2
19 > 2 ⇒ high = 19

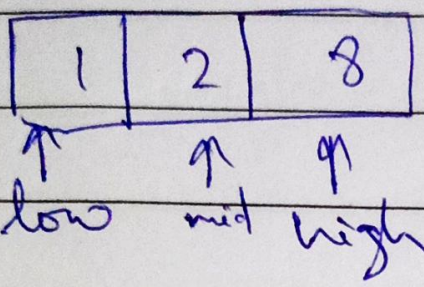
0	1	2	3	4
1	2	8	15	19

low ↑ (0) mid ↑ (2) high ↑ (4)

mid = $\frac{0+4}{2}$ = 2

1 ≠ 2 / 19 ≠ 2 / 8 ≠ 2

8 > 2 ⇒ high = 8



mid = 2

2 == 2 ,

We have found 2 at index 1.

Linear Search	Binary Search
- Works on both sorted and unsorted array	Works only on sorted array
- Equality operation	Inequality operation
- O(n) WC complexity	O(log n) WC complexity.