

Binary search

$n = 8$
 $arr[n] = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
 $key = 7$

Diagram illustrating the initial state of the array and the search process:

- s_1 points to index 1.
- s_2 points to index 5.
- e_1 points to index 8.
- 1st mid points to index 4.
- 2nd mid points to index 6.

```
int s = 0, e = n;
while (s <= e) {
    mid = (s + e) / 2;
    if (arr[mid] == key) {
        return mid;
    } else if (arr[mid] > key) {
        e = mid - 1;
    } else {
        s = mid + 1;
    }
}
```

Output

while ($0 \leq 8$)
 $mid = 8/2 = 4$ (1st mid)
else // $arr[mid] < key$ Now starting index is 5.
 $s = 4 + 1 = 5$
while ($5 \leq 8$)
 $mid = (5 + 8) / 2 = 13 / 2 = 6$ (2nd mid)
if ($arr[6] == 7$)
 $7 == 7$ ✓
return mid; // index of key element in array is returned.

Time complexity of Binary Search

After 1st itera, Length of ^{searching} array $\rightarrow n$

After 2nd itera, $\rightarrow n/2$

After 3rd itera, $\rightarrow (n/2)/2 = n/2^2 = \boxed{n/4}$

After k itera, $\rightarrow n/2^k$

Let the Length of array become 1 after k iteration:

$$n/2^k = 1$$

$$n = 2^k$$

$$\log_2(n) = \log_2(2^k)$$

$$\log_2(n) = k \boxed{\log_2 2} \rightarrow = 1$$
$$\Rightarrow \boxed{\log_2 n = k}$$

Time complexity:

$O(\log_2^n) \rightarrow$ Binary Search

$O(\frac{n}{1}) \rightarrow$ Linear search complexity