Searching in Array

Course on C-Programming & Data Structures: GATE - 2024 & 2025

Vishvadeep Gothi • Lesson 24 • Dec 23, 2022

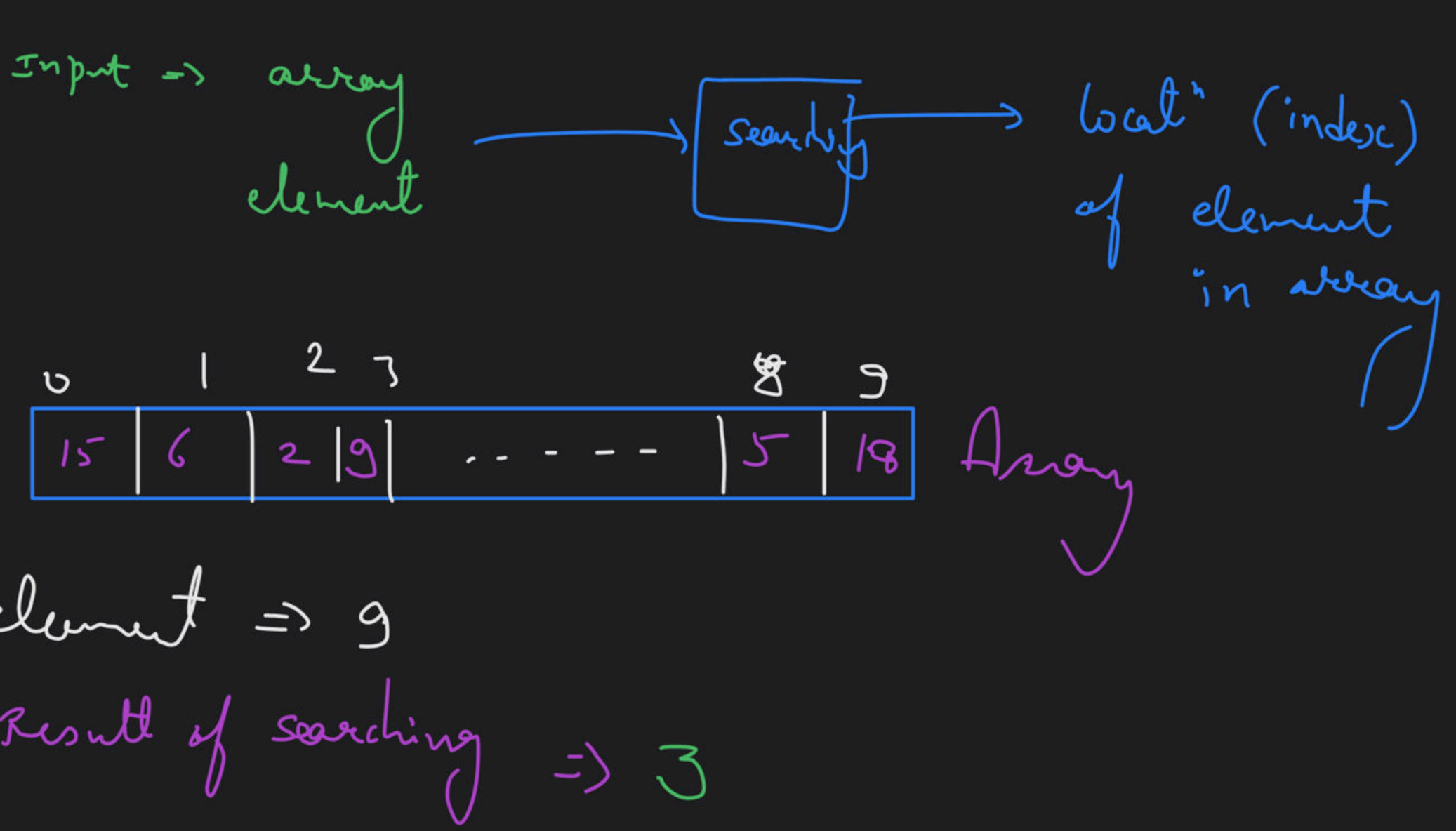
Data Structure:
Array: Searching

By: Vishvadeep Gothi

Hello!

I am Vishvadeep Gothi

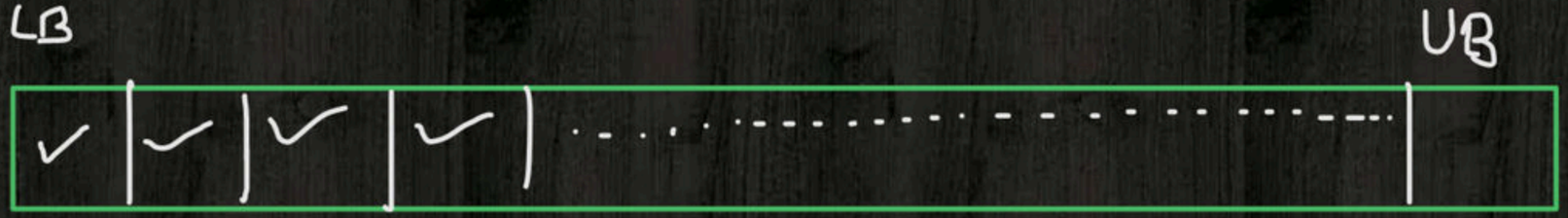
I am here because I love to teach


Input =>
array
element
search()
locat^n (index)
of element
in array
0
1
2
3
8
9
15
6
2
9
5
18
Array
element => 9
Result of searching => 3

Searching in Array

1. Linear Search
2. Binary Search

Linear Search




```
Linear_search(A[], LB, UB, element)
{
  for( k=LB; k<=UB; k++)
  {
    if( A[k] == element )
        return k;
  }
return LB-1;
}
```

R.T. Complexity	
Best case	$\Theta(1)$
worst case	$\Theta(n)$
avg case	$O(n)$

update linear search to find all indexes where given element is present.

```
Ret[] = {LB-1, ....., };
i = LB;
for(k = LB; k<=UB; k++)
{
  if (A[k] == element)
  {
    Ret[i] = k;
    i++;
  }}
return Ret;
```

Ret

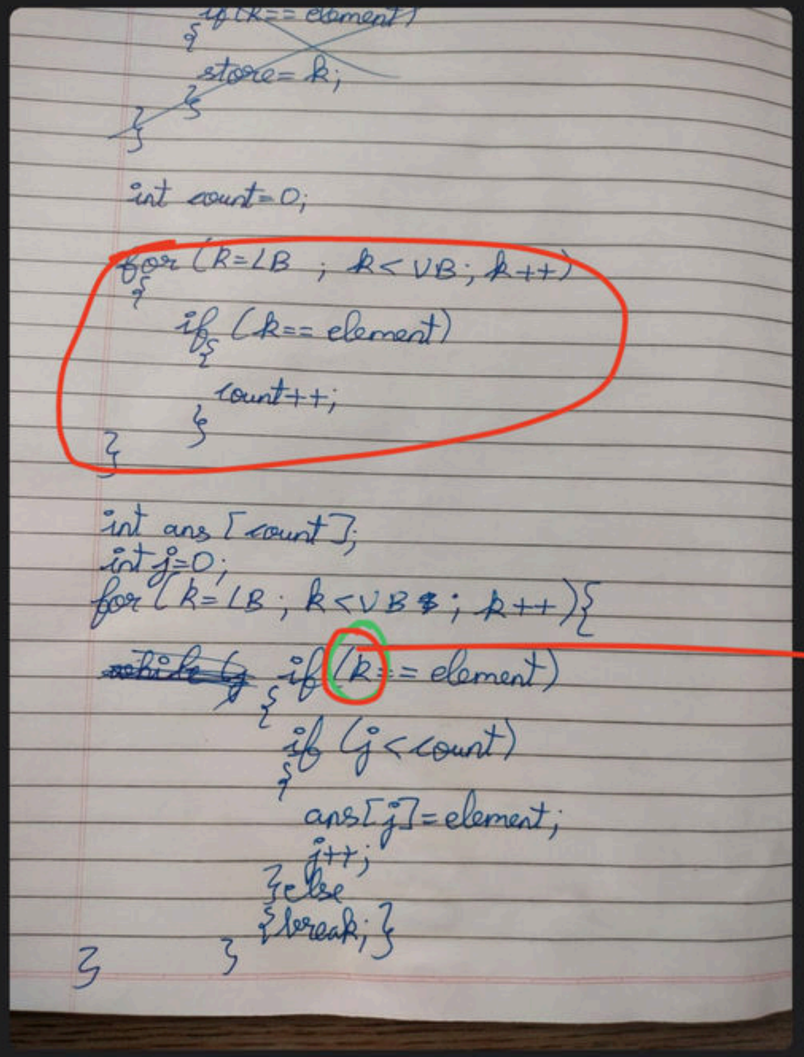
0	1	2		
3	10	15	-1	-1

R.T. Comp. $\Rightarrow \theta(n)$

unacademy

▲ 1 • Asked by Adhyay

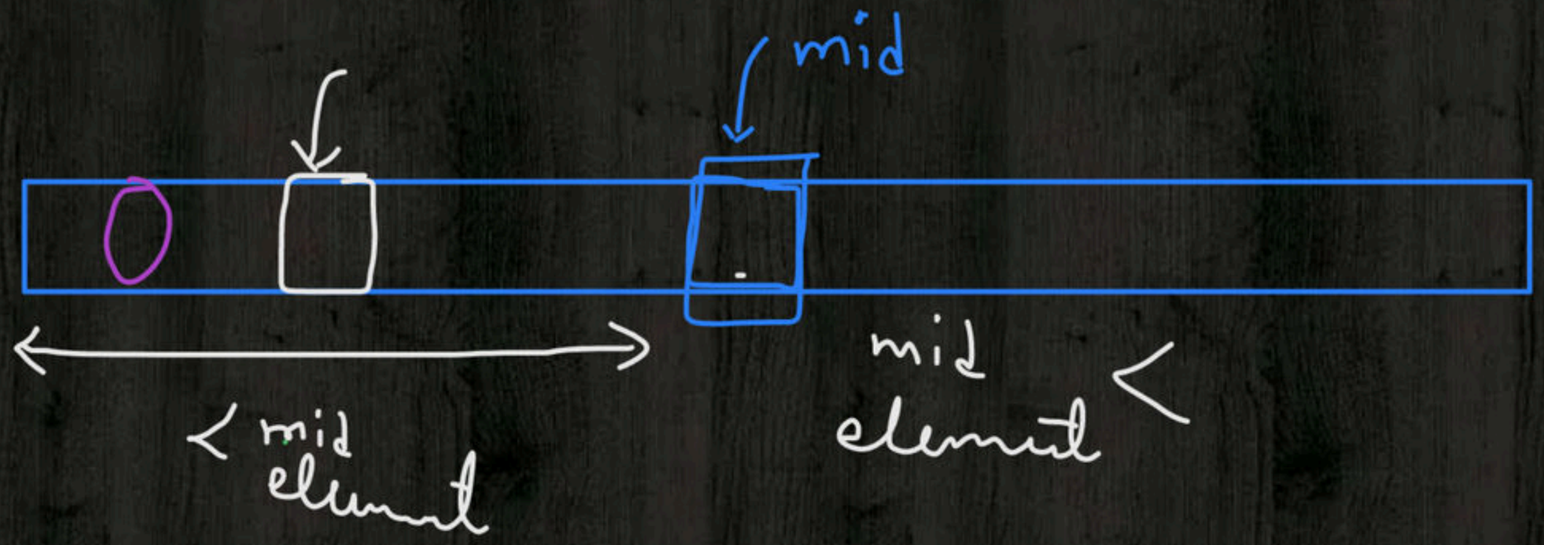
Please help me with this doubt

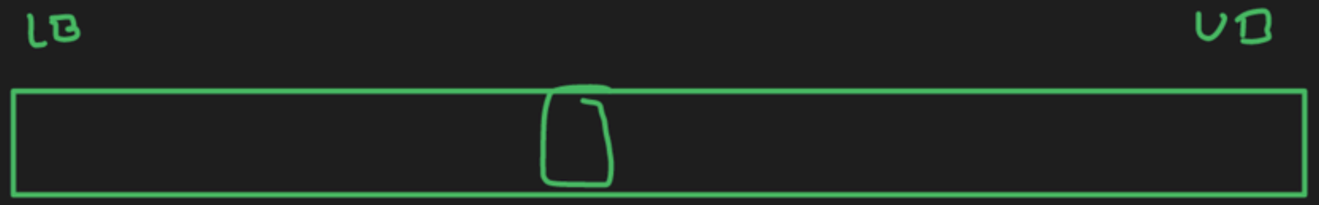


Binary Search

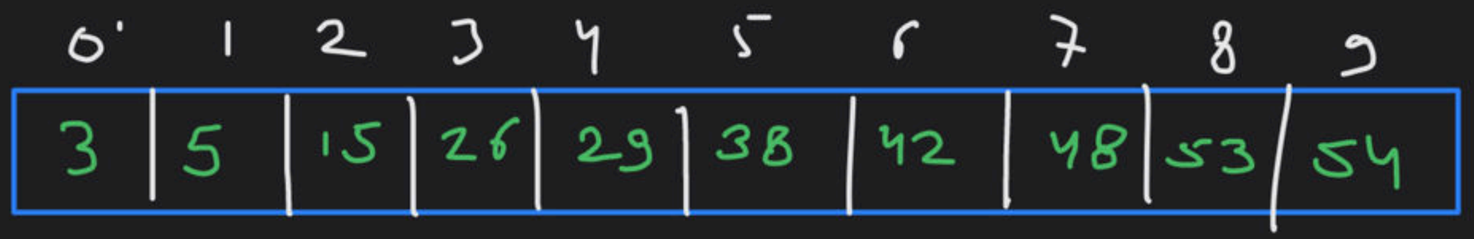
→ It works only on "sorted array"

$\frac{n}{2^4 8}$





$$\text{mid index} = \frac{LB + UB}{2}$$



search ⇒ 3

Low = 0

high = ~~9~~ 8 0

mid = ~~4~~ 1 0

LB = 0

UB = 9

if (item < A[mid]) ⇒ UB = mid − 1

Mid = $\frac{0+9}{2}$ = 4

if (item > A[mid]) ⇒ LB = mid + 1

Case 1:- search 48:-

LB = ~~0~~ 5

UB = 9

Mid = ~~4~~ $\frac{5+9}{2}$ = 7

A(7) == 48

return 7

case 2:- search 26

LB = ~~0~~ 2 3

UB = ~~8~~ 3

mid = ~~4~~ 1 2 3

return 3

Case 3:- Search 50

LB = ~~0~~ ~~5~~ 8

UB = ~~9~~ 7

mid = ~~4~~ 7 8

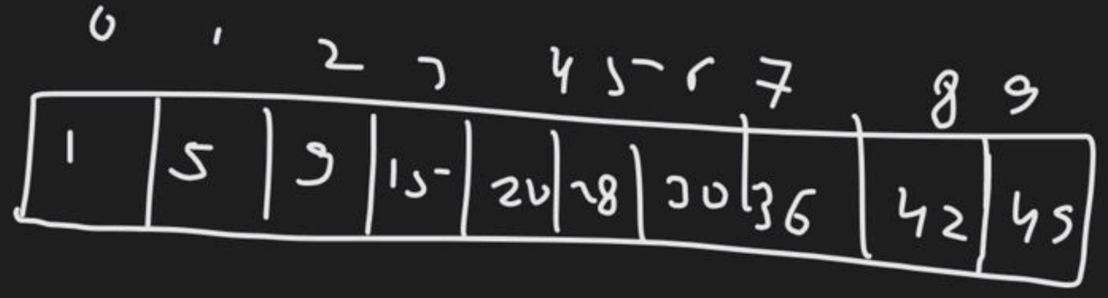
return −1

```
Binary-search (A[], item, LB, UB)
{
  Low = LB
  high = UB
  mid = (LB + UB)/2

  while (low <= high)
  {
    if (A[mid] == item)
      return mid;
    else if (item < A[mid])
      high = mid - 1;
    else
      low = mid + 1;
    mid = (low + high)/2;
  }
  return LB - 1;
}
```

```
while (A[mid] != item  && low <= High)
{
   if (item < A[mid])  high = mid - 1;
   else  low = mid + 1;
mid =(low+high)/2;
}
if(A[mid] == item)
    return  mid
else
    return LB - 1
```

R.T. Complexity = $O(\log_2 n)$



Search = 46

Low = 0 4 6 7 8 8 8 8 - - - -

High = 9

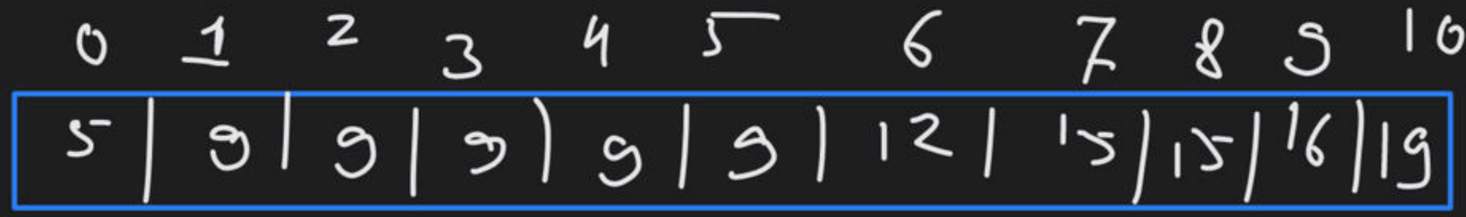
mid = 4 6 7 8 8 8 8 - - - -

Ques) Consider an unsorted array with duplicate elements.
R.T. complexity to find frequency of a given element k is this array of size n is ?

Ans:- $\theta(n)$

```
count = 0
for(i = LB to UB, i++)
{
  if(A[i] == k)
      count++;
}
return count;
```

Finding first & last appearance of a given element in sorted array :-



given element ⇒ 9

Return ⇒ 1 , 5

Question

Consider a sorted array of size n with duplicate elements. You have been given an element k, what is the time complexity to find the frequency of element k in the array?

(A) $\theta(1)$

(B) $\theta(\log n)$

(C) $\theta(n)$

(D) None

find first appearance of k in array $\Rightarrow$ i $\Rightarrow \log_2 n$

-"- last -"- $\Rightarrow$ j $\Rightarrow \log_2 n$

return (j - i + 1)

Total $\Rightarrow 2 * \log_2 n$

$\Rightarrow \log_2 n$

8:45 am

Question

Consider a sorted array of size n with duplicate elements. You have been given an element k, what is the time complexity to find that the element k is appeared atleast n/3 times or not?

(A) O (1)

(B) O (log n)

(C) O (n)

(D) None

Question

Consider a sorted array of size n with duplicate elements. Find the time complexity to find whether an elements is appeared more than n/2 times or not in the array?

(A) O (1)

(B) O (log n)

(C) O(n)

(D) None

Happy Learning