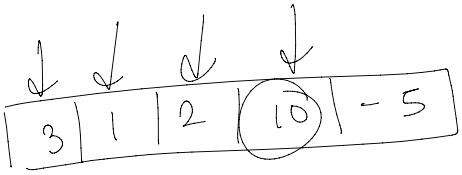
09 May 2024 21:39

Binary Search



K = 10

Linear Search

K, A

```
for(int i=0; i<A.size(); i++){
    if(A[i]==k){
        return true;
    }
}
```

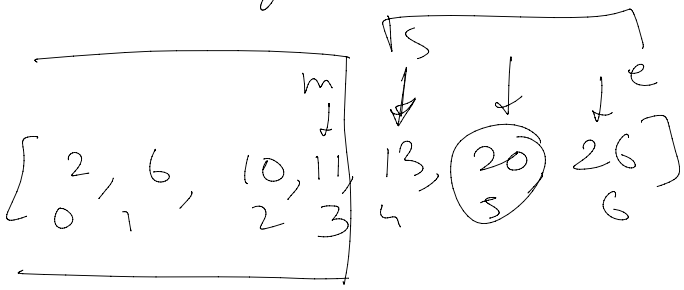
T: O(n)
S: O(1)

Binary

> Sorted array

> Unsorted array

> No array



K = 20

s e m

```
S=0
e=A.size()-1
m = S + (e-S)/2

if( A[m] < k){
    S = m+1;
}
```

S	e	m
0	6	3
4	6	5

```
S=0, e= A.size()-1, m;
while( s<=e){
    → m = s + (e-s)/2;
    if(A[m]==k){
        return true;
    }
    else if( A[m] > k){
        e = m-1;
    }
    else{
        s = m+1;
    }
}
```

* At any given point we should be able to decide whether we have to make start = mid+1 or end = mid-1.

T: $O(\log_2 n)$

$(2^4)\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{1}{2}\right)\left(\frac{1}{2}\right) = 1$

$(\log_2 n)$ $\log_2(2^4) \Rightarrow 4$

Square root of integer (Binary search on a search space)

Square root of integer (Binary ...

(1-25)

25

n
s=1 e=25

```
int ans = 0
while(s<=e){
    m = s + (e-s)/2;
    if(m*m==n){
        return m;
    }
    else if(m*m<n){
        ans=m;
        s=m+1;
    }
    else{
        e=m-1;
    }
}
return ans;
}
```

m=4 $\sqrt{20}$

34

s	e	m	ans
1	34	17	
1	16	8	
1	7	4	4
5	7	6	
5	5	5	5
6	5		

First and last occurrence of x

5

[1, 3, 5, 5, 5, 67, 123, 125]
0 1 2 3 4 5 6 7 8

[1, 3, (5), 5, (5), (5), 6, 12, 12, 12]
0 1 2 3 4 5 6 7 8

e = 7 3 4 / 5

I (Linear Search)

s = -1, e = -1; k

```
for(int i=0; i<A.size(); i++) {
    if(A[i] == k) {
        s = i;
        for(int j=i; j<A.size(); j++) {
            if(A[j] == k)
                e = j;
            }
        else
            break;
        }
    break;
}
```

T: O(n)

S: O(1)

Find first occurence

```
s=0; e=A.size()-1; k, m;
si=-1, ei=-1;
while(s<=e) {
    m = s + (e-s)/2;
    if(A[m] == k) {
        si = m;
        e = m-1;
```

s e
l m r
[1, 3, (5), 5, 5, 5, 6, 7, 12, 3, 12, 5]
0 1 2 3 4 5 6 7 8

si	s	e	m
4	0	8	4
	0	3	1
	2	3	2

```
        e=m-1;
    }
    else if(A[m]<k){
        s=m+1;
    }
    else{
        e=m-1;
    }
}
```

(2)

0	3	1
2	3	2
2	1	✗

Last Occurence

```
ei=-1
s=0, e=Asize()-1, m
while(s<=e){
    m=s+(e-s)/2;
    if(A[m]==k){
        ei=m;
        s=m+1;
    }
    else if(A[m]<k){
        s=m+1;
    }
    else{
        e=m-1;
    }
}
```

T: O(logn)

S: O(1)

find a peak element (Binary Search on unsorted array)

[1, 2, 3, 4]

[1, 20, 3, 12, 16]

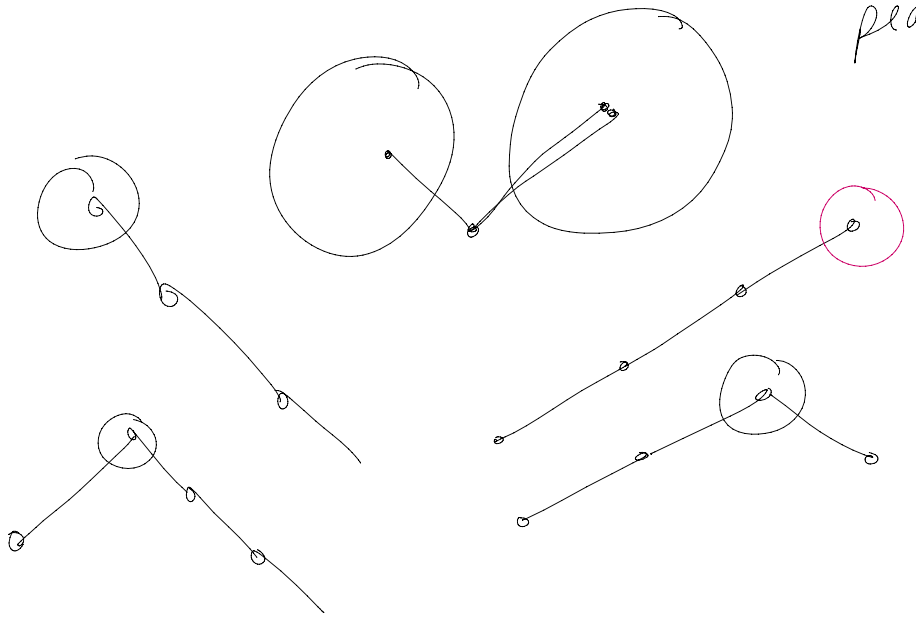
[1, 20, 2, 30, 4, 50, 6]

```
if (A.size()==1){
        return A[0];
}
if (A[0]>=A[1]){
        return A[0];
}
if (A[A.size()-1]>=A[A.size()-2]){
        return A[A.size()-1];
}
for (int i=1; i<A.size()-1; i++){
    if (A[i]>=A[i-1] && A[i]>=A[i+1]){
        return A[i];
    }
}
```

O(n) O(1)

1) Can we sort the given array?

peak = local maxima



1) start Binary search: $s=0$, $e=A.size()-1$.

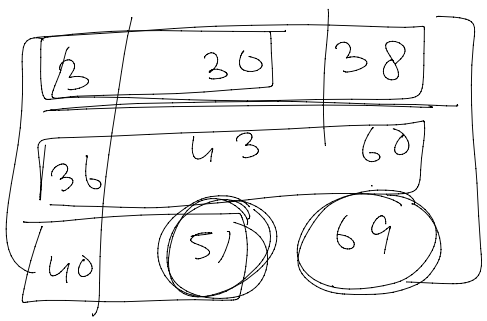
2) update $m = s + \frac{e-s}{2}$.

3) If $A[m]$ is peak, return it.

4) else if $A[m] < A[m+1]$, $s = m+1$.

5) else $e = m-1$.

Search in a row column sorted matrix

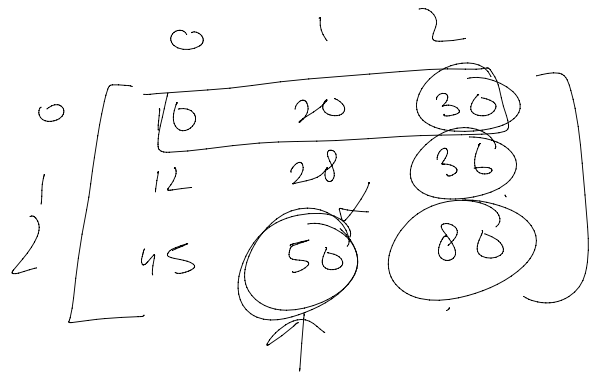


62

$r = 0 \quad 1 \quad 2 \quad 3$

$c = 2 \quad 1$

1) Binary search on each row.



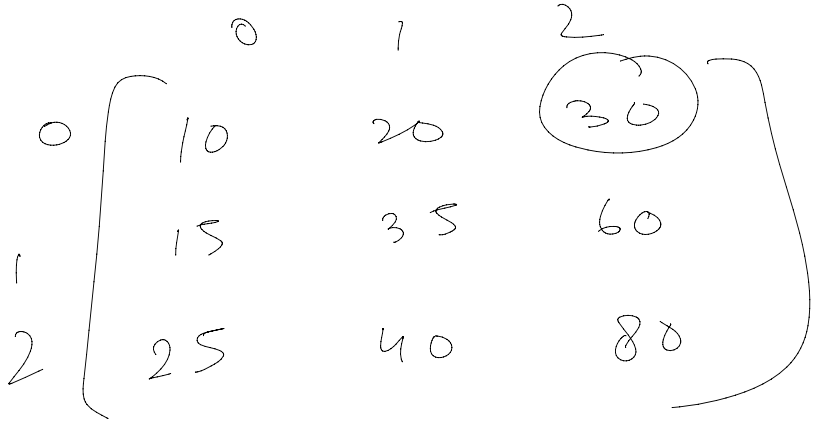
50

r = 0 1

c = 2

r++;

c--;



25

r = 0

c = 2 1