

## Binary Search →

(array is sorted in ascending order)


Steps →

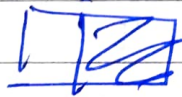
- ① find the middle element.
- ② if target > mid → search in right else ~~search~~ search in left.
- ③ if middle element == target element.


Time Complexity → Best Case  
O(1)


Worst Case →

 →  $N = \frac{N}{2^0}$

 →  $\frac{N}{2} = \frac{N}{2^1}$

 →  $\frac{N}{4} = \frac{N}{2^2}$

 →  $\frac{N}{8} = \frac{N}{2^3}$

 →  $\frac{N}{N} = \frac{N}{2^k}$

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

$\Rightarrow k = \log_2 N$

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

Time, worst case =  $O(\log N)$

$mid = \frac{start + end}{2} = \frac{l + (e-1)}{2}$

Mountain array, or Bitonic array  $\rightarrow$

$arr[0] < arr[1] < arr[2] < \dots < arr[i] > arr[i+1] > \dots$

$i$  = Mountain index

— If a sorted array is given in the problem statement, try BS.

— Ceiling = smallest element in an array greater than or equal to the target element.

— Floor = Biggest element in an array smaller than or equal to the target element.

Rotated Binary Search (RBS)

$arr = [2, 4, 5, 7, 8, 9, 10, 12]$

after 1 rotation  $\rightarrow$

$[12, 2, 4, 5, 7, 8, 9, 10]$

after 2 rotations  $\rightarrow$

$[10, 12, 2, 4, 5, 7, 8, 9]$

0 1 2 3 4 5 6 7

Start.

RBS with duplicate values  $\rightarrow$

arr = [2, 2, 2, 2, 9]

$\downarrow$  rotate once

[9, 2, 2, 2, 2]

$\downarrow$  rotate once more

[2, 9, 2, 2, 2]

① 1 ② 3 ④

s m e

$s = m = e$ , our normal RBS will fail.

We know s & e. & move one element to right or to left

[2, 9, 2, 2, 2]  
s m e

## \*\*\* Split Array

A given randomly sorted array is divided into  $m$  subarrays (all possible). Out of all the possibilities max sum of the elements of the subarray is taken and the minimum among all of these max sums is returned.

[7, 2, 5, 8, 10]

minSum = 10. (if  $m = 3$ )  
maxSum = 32 (if  $m = 1$ )

Range  $\rightarrow$  [10, 32]



$$\rightarrow \text{mid} = \frac{8+0}{2} = \frac{42}{2} = 21$$

try to see if you can split the array with 21 as the max sum

$$\rightarrow [7, 2, 5], [8, 10]$$

pieces  
2

if (pieces  $\leq$  m)

// It means that the individual sum must be less than 21

$$\rightarrow \text{end} = \text{mid} (= 21)$$

$$s = 10, \text{end} = 21$$

$$m = 5$$

$$[7, 2, 5], [8, 10]$$

(3)

(if pieces  $>$  m)

$$\rightarrow \text{Since } m = 2$$

$$s = \text{mid} + 1$$

$$s = 16, \text{end} = 21$$

$$m = 18$$

Pieces

$$[7, 2, 5], [8, 10]$$

2

$$s = 16, \text{end} = 18$$

$$m = 17$$

Pieces

$$\rightarrow [7, 2, 5], [8], [10]$$

3

$$n = 18$$

$$k = 18$$

$$m = 18$$

ans

Two dimensional Matrix Binary Search.

$$\text{Time complexity} = O(N^2)$$

for a sorted matrix.  $\rightarrow$  (sorted in row wise & column wise)  
we start searching from row = 0 & col = left.

Case 1 if element == target  
return target

Case 2 if element > target.  
col --

Case 3 if element < target  
row ++.

$$\text{Time complexity} = N + N = \underline{O(N)}$$

for strictly sorted Matrix  $\rightarrow$

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

$\rightarrow$  Take middle col & perform BS on it.

$\rightarrow$  if element == target  
ans

If element > target

Ignore rows below it.

If element < target

Ignore above rows

→ In the end, we'll be left with 2 rows  
In the 2 rows, check whether the middle  
col. has our element. otherwise,  
Consider the other four parts



Time complexity →

$O(\log(m) + \log(n))$  strategy

→ To shortlist

till 2 rows

→ Columning

~