

Problem: Given sorted (asc.) int array, return start + end index of given target [x, y]. Solution should be O(logn) time.

$$[3, 4, \boxed{8, 8, 8}, 10], 8 \Rightarrow [2, 4]$$

2 3 4

Intro:

- Verify Constraints
 -
- Create Testcases
 - [3, 4, 8, 8, 8, 9], 8 -> [2, 4]
 - [3, 4, 8, 8, 8], 6 -> [-1, -1]
 - [1] 1 -> [0, 0]
 - [] 8 -> [-1, -1]

Optimal 1:

- Brainstorming & Pattern Observations

```

ranges = [-1, 1]
if nums == None:
    return [-1, -1]

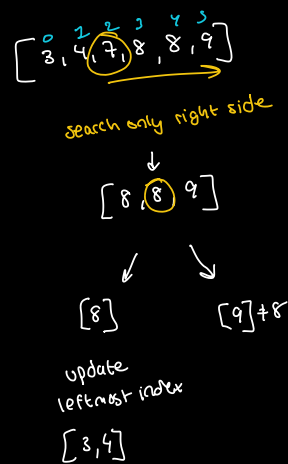
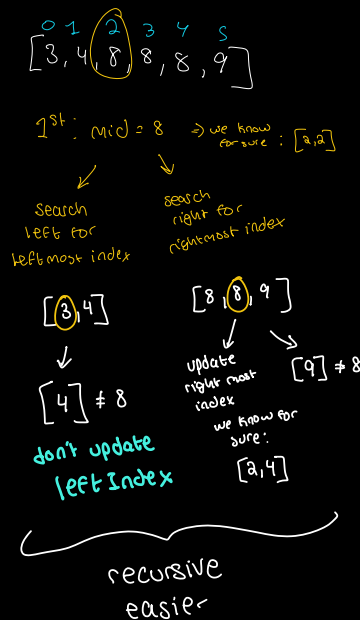
if len(nums) == 1:
    if nums[0] == target:
        return [0, 0]
    else:
        return [-1, -1]

if target < nums[0]
or nums[-1] < target:
    return [-1, -1]

prevMid = -1
  
```

```

when nums[mid] == target:
    if ranges[1] < mid:
        ranges[1] = mid
    if mid < ranges[0]:
        ranges[0] = mid
    if prevMid != -1: # right
        if prevMid < mid:
            bs(nums, target, mid+1, right, ranges, mid)
        else # left
            bs(nums, target, left, mid-1, ranges, mid)
    else:
  
```



⇒ regular binarySearch until find target

⇒ when find target

left search for leftmost index

right search for rightmost index

left

binarySearch (nums, target, low, mid-1, ranges, prevMid)
right
binarySearch (nums, target, mid+1, right, ranges, prevMid)

- Pseudocode
- Write code
- Run through testcases
- Analyze time and space complexity
 - Time: $O(\log n)$
 - Space: (recursive): $O(\log n)$

Iterative Optimal:
Time: $O(\log n)$
Space: $O(1)$