Search in Rotated Sorted Array

LeetCode Admin Jan 10, 2019

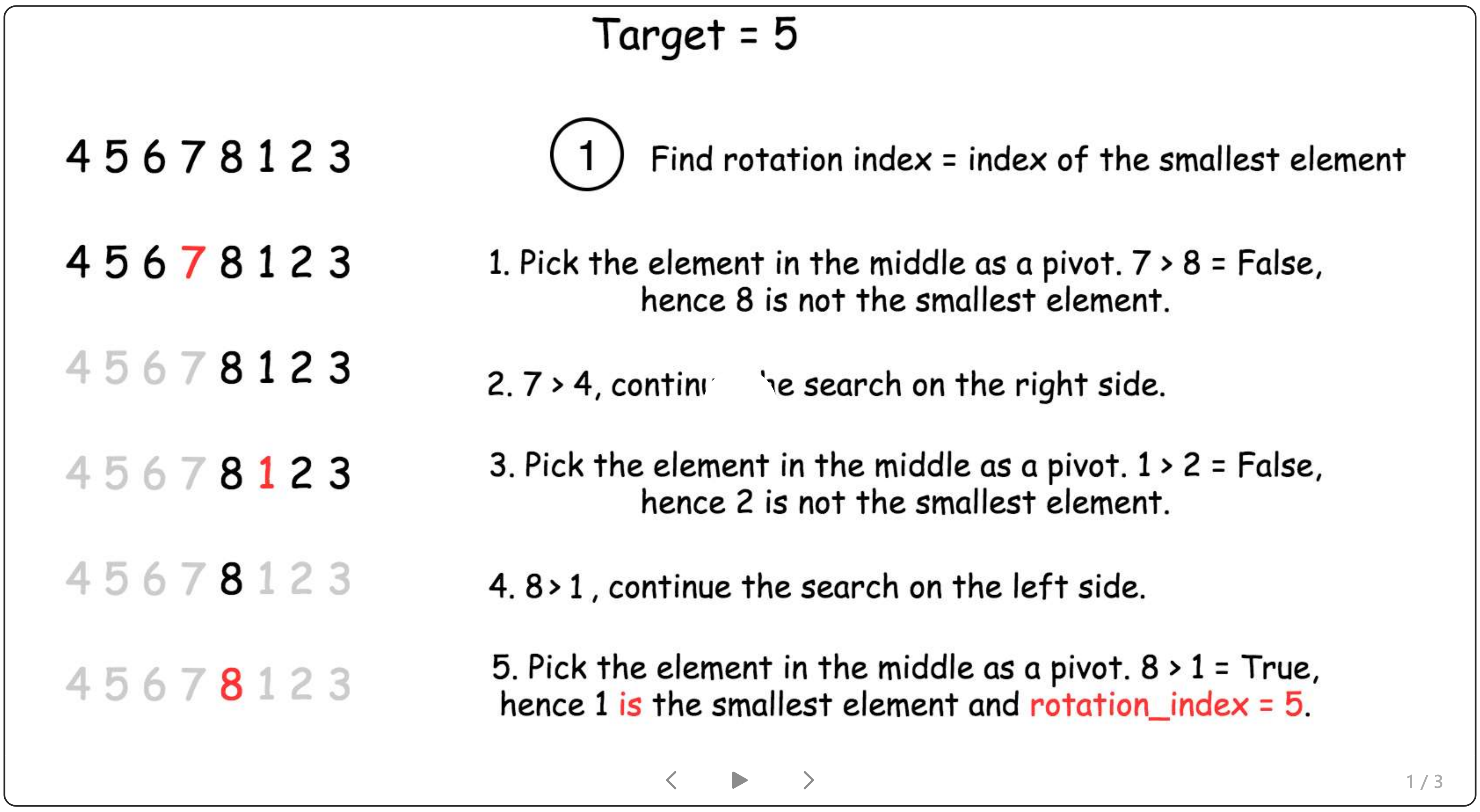
Solution

Approach 1: Binary search

The problem is to implement a search in $\mathcal{O}(\log N)$ time that gives an idea to use a binary search.

The algorithm is quite straightforward :

- Find a rotation index `rotation_index` , *i.e.* index of the smallest element in the array. Binary search works just perfect here.
- `rotation_index` splits array in two parts. Compare `nums[0]` and `target` to identify in which part one has to look for `target` .
- Perform a binary search in the chosen part of the array.



Java Python Copy

```java
class Solution {
  int [] nums;
  int target;

  public int find_rotate_index(int left, int right) {
    if (nums[left] < nums[right])
      return 0;

    while (left <= right) {
      int pivot = (left + right) / 2;
      if (nums[pivot] > nums[pivot + 1])
        return pivot + 1;
      else {
        if (nums[pivot] < nums[left])
          right = pivot - 1;
        else
          left = pivot + 1;
      }
    }
    return 0;
  }

  public int search(int left, int right) {
    /*
    Binary search
    */
    while (left <= right) {
```

Complexity Analysis

- Time complexity : $\mathcal{O}(\log N)$.
- Space complexity : $\mathcal{O}(1)$.

Approach 2: One-pass Binary Search

Instead of going through the input array in two passes, we could achieve the goal in one pass with an *revised* binary search.

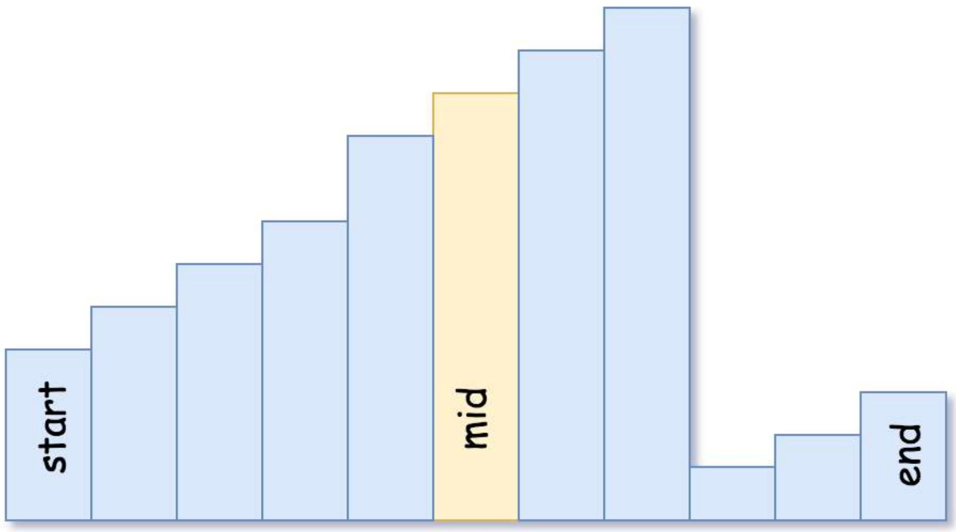
> The idea is that we add some additional *condition checks* in the normal binary search in order to better *narrow down* the scope of the search.

Algorithm

As in the normal binary search, we keep two pointers (*i.e.* `start` and `end`) to track the search scope. At each iteration, we reduce the search scope into half, by moving either the `start` or `end` pointer to the middle (*i.e.* `mid`) of the previous search scope.

Here are the detailed breakdowns of the algorithm:

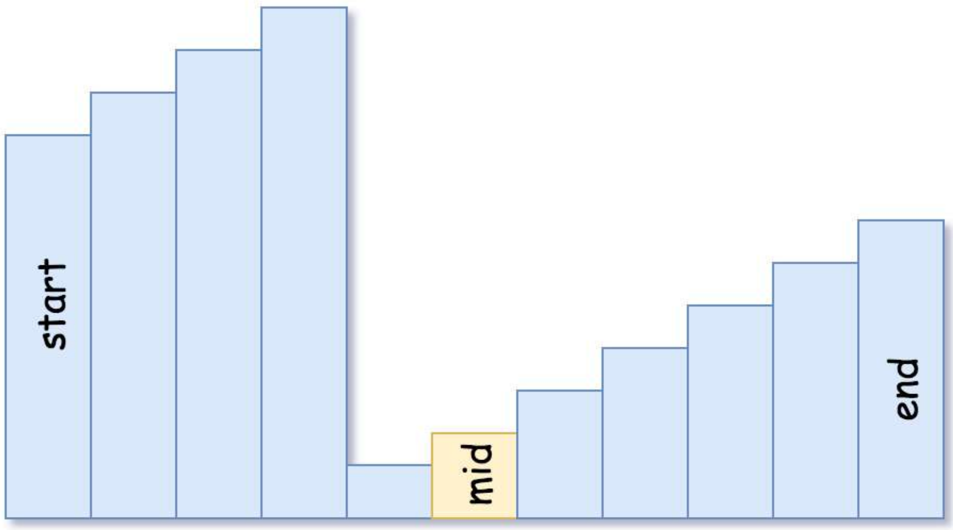
- Initiate the pointer `start` to `0`, and the pointer `end` to `n - 1`.
- Perform standard binary search. While `start <= end`:
  - Take an index in the middle `mid` as a pivot.
  - If `nums[mid] == target`, the job is done, return `mid`.
  - Now there could be two situations:
    - Pivot element is larger than the first element in the array, *i.e.* the subarray from the first element to the pivot is non-rotated, as shown in the following graph.




```
- If the target is located in the non-rotated subarray:
go left: `end = mid - 1`.

- Otherwise: go right: `start = mid + 1`.
```

- Pivot element is smaller than the first element of the array, *i.e.* the rotation index is somewhere between `0` and `mid`. It implies that the sub-array from the pivot element to the last one is non-rotated, as shown in the following graph.




```
- If the target is located in the non-rotated subarray:
go right: `start = mid + 1`.

- Otherwise: go left: `end = mid - 1`.
```

- We're here because the target is not found. Return -1.

Implementation

Java | Python

Copy

```java
class Solution {
  public int search(int[] nums, int target) {
    int start = 0, end = nums.length - 1;
    while (start <= end) {
      int mid = start + (end - start) / 2;
      if (nums[mid] == target) return mid;
      else if (nums[mid] >= nums[start]) {
        if (target >= nums[start] && target < nums[mid]) end = mid - 1;
        else start = mid + 1;
      }
      else {
        if (target <= nums[end] && target > nums[mid]) start = mid + 1;
        else end = mid - 1;
      }
    }
    return -1;
  }
}
```

Complexity Analysis

- Time complexity: $\mathcal{O}(\log N)$.
- Space complexity: $\mathcal{O}(1)$.

Share

Comments (162)

Sort by: **Best**

Type comment here... (Markdown supported)

Preview Comment

Article Commenting Rules

1. This comment section is for questions and comments regarding this **LeetCode article**. All posts must respect our LeetCode Community Rules.

2. Concerns about errors or bugs in the article, problem description, or test cases should be posted on LeetCode Feedback, so that our team can address them.

reyou Apr 27, 2019

Formula: If a sorted array is shifted, if you take the middle, always one side will be sorted. Take the recursion according to that rule.

1- take the middle and compare with target, if matches return.
2- if middle is bigger than left side, it means left is sorted
2a- if [left] < target < [middle] then do recursion with left, middle - 1 (right)
2b- left side is sorted, but target not in here, search on right side middle + 1 (left), right
3- if middle is less than right side, it means right is sorted
3a- if [middle] < target < [right] then do recursion with middle + 1 (left), right
3b- right side is sorted, but target not in here, search on left side left, middle -1 (right)

691 Show 23 Replies Reply

kakrafoon Jan 02, 2021

Took me a while to sort out the cases. I always get stuck figuring out whether to use '<=' or '<' in these type of problems. In this case, while the binary search itself was fine (<=), while looking for the pivot, I used '<'. Need to practice until this becomes second nature I think. Either that or just fix (<=) as the convention and take it from there.

109 Show 5 Replies Reply

sutirtho Jun 25, 2020

It is an moderately easy concept but can be very tricky to code.

144 Show 1 Replies Reply

lim142857 Jul 27, 2019

Python 3 modified binary search (readable)(fast)(short)

```
    # Initilize two pointers
    begin = 0
    end = len(nums) - 1
    while begin <= end:
        mid = (begin + end)//2
        if nums[mid] == target:
            return mid
        if nums[mid] > nums[end]: # Left side of mid is sorted
```

74 Show 4 Replies Reply

redcoder7 Jul 14, 2021

Read more

38 Show 5 Replies Reply

haoyangfan Jan 25, 2019

Read more

68 Show 8 Replies Reply

Sithis Jan 11, 2019

Read more

41 Show 12 Replies Reply

softwareshortcut Apr 19, 2019

Creating helper methods is key otherwise you'd easily get lost dealing with indices. Great exercise, thanks for sharing. I see shorter solutions in the comments, but they might be tricky to come up with during an interview.

51 Reply

layak Jun 06, 2020

why do we have equal to sign in this condition? nums[mid] >= nums[start]

23 Show 7 Replies Reply

rubalwanchoo Apr 03, 2019

In one of the test case, the input array is [1,3], target 0
How is [1,3] a rotated array?

17 Show 7 Replies Reply

1 2 3 4 5 6 ... 17