**Viral Shaileshkumar Jariwala**

Question 1 :

33. Search in Rotated Sorted Array

Medium

20K

1.2K

Companies

There is an integer array nums sorted in ascending order (with **distinct** values).

Prior to being passed to your function, nums is **possibly rotated** at an unknown pivot index k (1 <= k < nums.length) such that the resulting array is [nums[k], nums[k+1], ..., nums[n-1], nums[0], nums[1], ..., nums[k-1]] (**0-indexed**). For example, [0,1,2,4,5,6,7] might be rotated at pivot index 3 and become [4,5,6,7,0,1,2].

Given the array nums **after** the possible rotation and an integer target, return *the index of*target*if it is in*nums*, or*-1*if it is not in*nums.

You must write an algorithm with O(log n) runtime complexity.

**Example 1:**

**Input:** nums = [4,5,6,7,0,1,2], target = 0

**Output:** 4

**Example 2:**

**Input:** nums = [4,5,6,7,0,1,2], target = 3

**Output:** -1

**Example 3:**

**Input:** nums = [1], target = 0

**Output:** -1

Answer :

Java :

class Solution {

   public int search(int[] nums, int target) {

        if (nums == null || nums.length == 0) {

            return -1;

        }

        int left = 0;

        int right = nums.length - 1;

        while (left < right) {

            int middle = left + (right - left) / 2;

            if (nums[middle] > nums[right]) {

                left = middle + 1;

            }

            else {

                right = middle;

            }

        }

        int pivot = left;

        left = 0;

        right = nums.length - 1;

        if (target >= nums[pivot] && target <= nums[right]) {

            left = pivot;

        } else {

            right = pivot;

        }

        while (left <= right) {

            int middle = left + (right - left) / 2;

            if (nums[middle] == target) {

                return middle;

            } else if (target < nums[middle]) {

                right = middle - 1;

            } else {

                left = middle + 1;

            }

        }

        return -1;

    }

}

Output :

