33. Search in Rotated Sorted Array

Description Hints Submissions Discuss Solution

Pick One

Suppose an array sorted in ascending order is rotated at some pivot unknown to you beforehand.

(i.e., `[0,1,2,4,5,6,7]` might become `[4,5,6,7,0,1,2]`).

You are given a target value to search. If found in the array return its index, otherwise return `-1` .

You may assume no duplicate exists in the array.

Your algorithm's runtime complexity must be in the order of *O*(log *n*).

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
```

Seen this question in a real interview before? Yes No

```java
public class L33 {

    /*
     * 这是在一个旋转有序数组里面查找目标元素的问题。
     * 问题解决思路：利用二分法来求解
     * 比如nums = [4,5,6,7,0,1,2], target = 0
     * 当二分法找
     *   mid = 3;
     *     nums[mid] = 7不等于0，由于7>2,进入第三个分支
     *   此时，nums[0] = 4 > 0
     *   所以left = 4。
     *   进入下一个循环。
     */
    public int search(int [] nums, int target) {
        if(nums.length == 0)
            return -1;
        int left = 0, right = nums.length - 1;
        while (left <= right) {     //重点注意这块地方是<=
            int mid = (left + right) / 2;
            if (nums[mid] == target) {
                return mid;
            }else if(nums[mid] < nums[right]){
                if (nums[mid] < target && nums[right] >= target) { //对于target来说都有=
                    left = mid + 1;
                }else {
                    right = mid - 1;
                }
            }else {
                if(nums[left] <= target && nums[mid] > target)
                    right = mid - 1;
                else {
                    left = mid + 1;
                }
            }
        }
        return -1;
    }
}
```