33. Search in Rotated Sorted Array

Created	@July 16, 2020 2:18 AM
Difficulty	Medium
LC Url	https://leetcode.com/problems/search-in-rotated-sorted-array/
Importance	****
Tag	Binary search NEET
Video	https://www.youtube.com/watch?v=7XOQIMZVIjA

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
```

Constraints:

- `1 <= nums.length <= 5000`
- `10 4 <= nums[i] <= 10 4`
- All values of `nums` are **unique**.
- `nums` is an ascending array that is possibly rotated.
- `10 4 <= target <= 10 4`

Solution

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
        left, right = 0, len(nums) - 1
        while left <= right:
            mid = (left + right) // 2
            if nums[mid] == target:
                return mid

            if target >= nums[0]:
                if nums[mid] >= nums[0] and target > nums[mid]:
                    left = mid + 1
                else:
                    right = mid - 1
            else:
                if nums[mid] >= nums[0] or target > nums[mid]:
                    left = mid + 1
                else:
                    right = mid - 1
        return -1
```