Search in Rotated Sorted Array

Difficulty	Medium
Category	Binary Search
Question	https://leetcode.com/problems/search-in-rotated-sorted-array/
Solution	https://youtu.be/U8XENwh8Oy8
Status	In progress

Question

> 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`.

Example

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
```

Idea

Find the pivot to divide into two half, check which half the target is at and do binary search on that half

Solution

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
        l, r = 0, len(nums) - 1

        while l <= r:
            mid = (l + r) // 2
            if target == nums[mid]:
                return mid

            # left sorted portion
            if nums[l] <= nums[mid]:
                if target > nums[mid] or target < nums[l]:
                    l = mid + 1
                else:
                    r = mid - 1
            # right sorted portion
            else:
                if target < nums[mid] or target > nums[r]:
                    r = mid - 1
                else:
                    l = mid + 1
        return -1
```

Explanation