

# 33. Search in Rotated Sorted Array

## Description

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` ( $1 \leq k < \text{nums.length}$ ) 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:

- $1 \leq \text{nums.length} \leq 5000$
- $-10^4 \leq \text{nums}[i] \leq 10^4$
- All values of `nums` are **unique**.
- `nums` is an ascending array that is possibly rotated.
- $-10^4 \leq \text{target} \leq 10^4$

