


Search in Rotated Sorted Array

# Index	33
📅 CreatedAt	@September 28, 2022
👤 Person	 Ally Hyeseong Kim
🌟 Status	Done
🏷️ Tags	Binary Search Python
📅 UpdatedAt	@September 28, 2022

References


LeetCode - The World's Leading Online Programming Learning Platform
Level up your coding skills and quickly land a job. This is the best place to expand your knowledge and get prepared for your next interview.

 <https://leetcode.com/problems/search-in-rotated-sorted-array/>



파이썬 알고리즘 인터뷰

2021 세종도서 학술부문 선정작. 현업과 실무에 유용한 주요 알고리즘 이론을 깊숙이 이해하고, 파이썬의 핵심 기능과 문법까지 상세하게 이해할 수 있는 취업용 코딩 테스트를 위한 완벽 가이드다. 200여 개가 넘는...

 <https://www.aladin.co.kr/shop/wproduct.aspx?ItemId=245495826>



References

1. 정렬 후 Binary Search
2. 정렬 없이 Binary Search

1. 정렬 후 Binary Search

```
class Solution:
    def search(self, nums: List[int], target: int) -> int:
        k = 0
        for i in range(1, len(nums)):
            if nums[i - 1] > nums[i]:
                k = i
                nums = nums[i:] + nums[:i]
                break

        left = 0
        right = len(nums) - 1
        index = (left + right) // 2
        while True:
            if left > right:
                return -1

            if nums[index] == target:
                return (k + index) % len(nums)
            elif nums[index] > target:
                right = index - 1
                index = (left + right) // 2
            else:
                left = index + 1
                index = (left + right) // 2
```

2. 정렬 없이 Binary Search

```
class Solution:
    def permute(self, nums: List[int]) -> List[List[int]]:
        if not nums:
            return -1

        left = 0
        right = len(nums) - 1
        while left < right:
            mid = left + (right - left) // 2

            if nums[mid] > nums[right]:
                left = mid + 1
            else:
                right = mid
        pivot = left

        left = 0
        right = len(nums) - 1
        while left <= right:
            mid = left + (right - left) // 2
            mid_pivot = (mid + pivot) % len(nums)

            if nums[mid_pivot] < target:
                left = mid + 1
            elif nums[mid_pivot] > target:
                right = mid - 1
            else:
                return mid_pivot
        return -1
```