二分查找

```python
# 1. 有序数组
def base_search(nums, target):
    '''
    实现一个基础二分查找
    输入: 一个顺序list
    输出: 待查找的元素的位置
    '''
    left, right = 0, len(nums) - 1
    while left <= right:
        mid = left + (right - left) // 2  # (left + right) >> 1 除2 向下取整
        if nums[mid] == target:
            return mid
        elif target < nums[mid]:
            right = mid - 1
        elif target > nums[mid]:
            left = mid + 1
    return -1


def left_search(nums, target):
    '''
    存在多个相同数字
    左边界
    '''
    left, right = 0, len(nums) - 1
    while left <= right:
        mid = (left + right) // 2
        if nums[mid] == target:
            right = mid - 1
        elif nums[mid] < target:
            left = mid + 1
        elif nums[mid] > target:
            right = mid - 1
    if left == len(nums) or nums[left] != target:
        return -1
    return left

def right_search(nums, target):
    '''
    右边界
    '''
    left, right = 0, len(nums) - 1
    while left <= right:
        mid = (left + right) // 2
        if nums[mid] == target:
            left = mid + 1
        elif nums[mid] < target:
            left = mid + 1
        elif nums[mid] > target:
            right = mid - 1
    if right < 0 or nums[right] != target:
        return -1
    return right
```

```python


# 因为我们初始化 right = len(nums) - 1
# 所以决定了我们的「搜索区间」是 [left, right]
# 所以决定了 while (left <= right)
# 同时也决定了 left = mid + 1 和 right = mid - 1


# 2. 旋转有序数组
def search(nums, target):
    '''
        在数组中搜索 target 值
        [2,5,6,0,0,1,2]
    '''
    left, right = 0, len(nums) - 1
    while left <= right:
        mid = (left + right) // 2
        if nums[mid] == target: return True
        if nums[mid] < nums[right]: # mid 在右半部分
            if target > nums[mid] and target <= nums[right]: # target 在右半部分
                left = mid + 1
            else: # target 在左半部分
                right = mid - 1

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

        elif nums[mid] == nums[right]: # 无法判断 存在相同元素
            right -= 1

    return False
```