leetcode33.搜索旋转排序数组——中等

笔记本：	leetcode刷题		
创建时间：	2021/8/9 23:11	**更新时间：**	2021/8/12 23:45
作者：	Zard		

整数数组nums按升序排列，数组中的值互不相同。
在传递给函数之前，nums在预先未知的某个下标k(0<=k<nums.length)上进行了旋转，使数组变为[nums[k],nums[k+1],...,nums[n-1],nums[0],nums[1],...,nums[k-1]](下标从0开始计数)。例如，[0,1,2,4,5,6,7]在下标3处经旋转后可能变为[4,5,6,7,0,1,2]。
给你旋转后的数组nums和一个整数target，如果nums中存在这个目标值target，则返回它的下标，否则返回-1。

示例 1：
输入：nums = [4,5,6,7,0,1,2], target = 0
输出：4

示例 2：
输入：nums = [4,5,6,7,0,1,2], target = 3
输出：-1

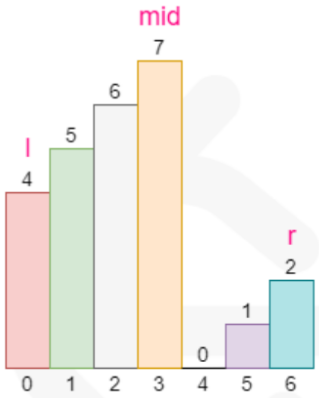
示例 3：
输入：nums = [1], target = 0
输出：-1

提示：
1 <= nums.length <= 5000
-10^4 <= nums[i] <= 10^4
nums 中的每个值都 独一无二
题目数据保证 nums 在预先未知的某个下标上进行了旋转
-10^4 <= target <= 10^4

进阶：你可以设计一个时间复杂度为 O(log n) 的解决方案吗?

方法一：二分查找
思路和算法：
对于有序数组，可以使用二分查找的方法查找元素。
我们发现的是，我们将数组从中间分开成左右两部分的时候，一定有一部分的数组是有序的。
针对本题我们可以在常规二分查找的时候查看当前mid为分割位置分割出来的两个部分[1,mid]和[mid+1,r]哪个部分是有序的，并根据有序的那个部分确定我们该如何改变二分查找的上下界，因为我们能够根据有序的那部分判断出target在不在这个部分：

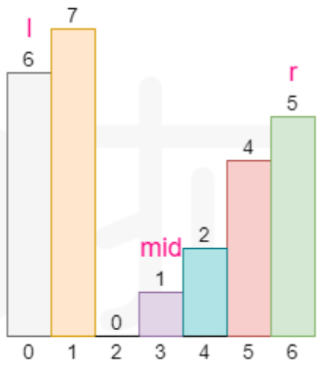
- 如果[1,mid-1]是有序数组，且target的大小满足[nums[l],nums[mid]],则我们应该将搜索范围缩小至[l,mid-1],否则在[mid+1,r]中寻找。
- 如果[mid,r]是有序数组，且target的大小满足（nums[mid+1],nums[r]），则我们应该将搜索范围缩小至[mid+1,r],否则在[l,mid-1]中寻找。



[l, mid] 是有序数组

如果 target = 5，在 [l, mid - 1] 中寻找

如果 target = 2，在 [mid + 1, r] 中寻找



[mid + 1, r] 是有序数组

如果 target = 6，在 [l, mid - 1] 中寻找

如果 target = 4，在 [mid + 1, r] 中寻找

```
class Solution {
public:
    int search(vector<int>& nums, int target) {
        int n=nums.size();
        if(!n){
            return -1;
        }
        if(n==1){
            return nums[0]==target?0:-1;
        }
        int l=0,r=n-1;
        while(l<=r){
            int mid=(l+r)/2;
            if(nums[mid]==target)    return mid;
            if(nums[0]<=nums[mid]){  //左边部分有序
                if(nums[0]<=target&&target<nums[mid])   //目标值在有序中
                    r=mid-1;
                else
                    l=mid+1;
            }else{   //右边部分有序
                  if(nums[mid]<target&&target<=nums[n-1])
                    l=mid+1;
                  else
                    r=mid-1;
                }
        }
        return -1;
    }
};
```