33. Search in Rotated Sorted Array

33. Search in Rotated Sorted Array

没太看懂题目, 想多了, 实际上只有两个降序的数组, 两次二分就行了.

```
public int search(int[] nums, int target) {
    if(nums.length == 1) return nums[0] == target ? 0 : -1;
    if(nums[nums.length - 1] < nums[0]){
        int index = findIndex(nums,target);
        int res =  binarySearch(nums,target, 0, index);
        return res >= 0 ? res : binarySearch(nums, target, index + 1,
nums.length - 1);
    }
    return binarySearch(nums, target, 0, nums.length - 1);

}

public int findIndex(int[] nums, int target){
    int start = 0, end = nums.length -1;
    int mid;
    while(start < end){
        mid = start + (end - start)/2;
        if(nums[mid] > nums[start]){
            if(nums[mid + 1] < nums[mid]) return mid;
            start = mid + 1;
        }else if(nums[mid] < nums[start]){
            if(nums[mid - 1] > nums[mid]) return mid - 1;
            end = mid - 1;
        }else{
            return start;
        }
    }
    return -1;
}

public int binarySearch(int[] nums, int target, int start, int end){
    int mid;
    while(start <= end){
        mid = start + (end - start) / 2;
        if(nums[mid] == target){
            return mid;
        }else if(nums[mid] > target){
            end = mid - 1;
        }else{
            start = mid + 1;
        }
    }
    return -1;
}
```

这块得弄清楚为什么上面一个while是start<=end(查找值)

下面是start < end并且下面end = mid 不是end = mid - 1;

```
public int search(int[] nums, int target) {
    int minIdx = findMinIdx(nums);
    if (target == nums[minIdx]) return minIdx;
    int m = nums.length;
    int start = (target <= nums[m - 1]) ? minIdx : 0;
    int end = (target > nums[m - 1]) ? minIdx : m - 1;

    while (start <= end) {
        int mid = start + (end - start) / 2;
        if (nums[mid] == target) return mid;
        else if (target > nums[mid]) start = mid + 1;
        else end = mid - 1;
    }
    return -1;
}

public int findMinIdx(int[] nums) {
    int start = 0, end = nums.length - 1;
    while (start < end) {
        int mid = start + (end -  start) / 2;
        if (nums[mid] > nums[end]) start = mid + 1;
        else end = mid;
    }
    return start;
}
```

最简单的

```
    public int search(int[] nums, int target) {
        int start = 0, end = nums.length - 1;
        while(start <= end){
            int mid = start + (end - start) / 2;
            if(nums[mid] == target) return mid;
            if(nums[start] > nums[mid]){
                if(nums[mid] > target || nums[end] < target){
                    end = mid - 1;
                }else{
                    start = mid + 1;
                }
            }else{//right rotate
                if(target > nums[mid] || target < nums[start])
                    start = mid + 1;
                else{
                    end = mid - 1;
                }
            }
        }
        return -1;
    }
```

81. Search in Rotated Sorted Array II

link

这块得分清，上面是left rotate所以跟nums[end]比，这样能确定在左边或者右边，下面是right rotate所以得跟nums[start]比

```
public boolean search(int[] nums, int target) {
    int start = 0, end = nums.length - 1;
    while(start <= end){
        int mid = start + (end - start) / 2;
        if(nums[mid] == target) return true;
        if(nums[start] > nums[mid]){
            if(nums[mid] > target || nums[end] < target){
                end = mid - 1;
            }else{
                start = mid + 1;
            }
        }else if(nums[start] < nums[mid]){//right rotate
            if(target > nums[mid] || target < nums[start])
                start = mid + 1;
            else{
                end = mid - 1;
            }
        }else{
            start++;
        }
    }
    return false;
}
```