**难度: 简单**

给定一个排序数组和一个目标值，在数组中找到目标值，并返回其索引。如果目标值不存在于数组中，返回它将会被按顺序插入的位置。

你可以假设数组中无重复元素。

示例 1:

输入: [1,3,5,6], 5

输出: 2

示例 2:

输入: [1,3,5,6], 2

输出: 1

示例 3:

输入: [1,3,5,6], 7

输出: 4

示例 4:

输入: [1,3,5,6], 0

输出: 0

来源：力扣（LeetCode）

链接：https://leetcode-cn.com/problems/search-insert-position

著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。

**思路1:**

1). 下标为0的时候就判断是否大于等于下标为0的,如果大于等于直接返回下标0

2). 但下标为0不成立的时候,直接比较是否大于并且比较下一个下标是否小于或等于

3). 如果可以持续循环下去,此时说明它是整个数组中最大的,因此只需要返回数组的长度当成它的下标即可

|  |
| --- |
| public int searchInsert(int[] nums, int target) {  if (nums[0] >= target) {  return 0;  }  for (int i = 0; i < nums.length - 1; i++) {  if (nums[i] < target && nums[i+1] >= target) {  return i+1;  }  }  return nums.length;  } |

**思路2(二分查找):**

1). 先定义左右指针

2). 然后使用左右指针的中间节点来进行比较

3). 如果相等直接返回下标

4). 如果小于的话,那么左指针,向右边移动mid+1

5). 如果大于的话,那么右指针,向左移动mid-1

6). 如果没找到直接返回l

|  |
| --- |
| public int searchInsert2(int[] nums, int target) {  int l = 0;  int r = nums.length -1;  while(l <= r) {  int mid = (l+r)/2;  if (nums[mid] == target) {  return mid;  }else if (nums[mid] < target) {  l = mid + 1;  }else {  r = mid - 1;  }  }  return l;  } |