**难度: 简单**

统计一个数字在排序数组中出现的次数。

示例 1:

输入: nums = [5,7,7,8,8,10], target = 8

输出: 2

示例 2:

输入: nums = [5,7,7,8,8,10], target = 6

输出: 0

限制：

0 <= 数组长度 <= 50000

来源：力扣（LeetCode）

链接：https://leetcode-cn.com/problems/zai-pai-xu-shu-zu-zhong-cha-zhao-shu-zi-lcof

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). 定义一个计数的变量

2). 循环这个数组

3). 因为这是一个排好序的数组,所以如果这个数组的数 大于 我们要查找的 值,直接结束循环

4). 如果相同就计数变量++;

|  |
| --- |
| public int search(int[] nums, int target) {  int count = 0; //出现次数  for (int i = 0; i < nums.length; i++) {  if (nums[i] > target) {  break;  }  if (nums[i] == target) {  count++;  }  }  return count;  } |

**思路2:**

1). 先寻找到最右边的点

2). 在寻找到最左的点

3). 然后把最右边的点 减 最左边的点 -1

为什么减1

因为最右边的点不是要找的值,而是最右边的上一个点才是,所以要减1

|  |
| --- |
| public int search2(int[] nums, int target) {  if(nums.length <= 0) {  return 0;  }  int tempLeft = 0;  int tempRight = nums.length - 1;  while(tempLeft <= tempRight) {  int mid = (tempLeft + tempRight) / 2;  if (nums[mid] <= target) {  tempLeft = mid+1;  }else {  tempRight = mid - 1;  }  }  //把最右点给到右  int right = tempLeft;  //重新赋值  tempLeft = 0;  tempRight = nums.length - 1;  while (tempLeft <= tempRight) {  int mid = (tempLeft + tempRight) / 2;  if (nums[mid] < target) {  tempLeft = mid+1;  }else {  tempRight = mid - 1;  }  }  //把最左点给到左  int left = tempRight;  return right - left -1;  } |