二分查找

二分查找总结

link 写的很好

标准二分查找

```
class BinarySearch {
    public int search(int[] nums, int target) {
        int left = 0;
        int right = nums.length - 1;
        while (left <= right) {
            int mid = left + ((right - left) >> 1);
            if (nums[mid] == target) return mid;
            else if (nums[mid] > target) {
                right = mid - 1;
            } else {
                left = mid + 1;
            }
        }
        return -1;
    }
}
```

二分查找左边界

左边界查找的第二种类型用于数组部分有序且包含重复元素的情况，这种条件下在我们向左收缩的时候，不能简单的令 right = mid，因为有重复元素的存在，这会导致我们有可能遗漏掉一部分区域，此时向左收缩只能采用比较保守的方式，代码模板如下

```
class Solution {
    public int search(int[] nums, int target) {
        int left = 0;
        int right = nums.length - 1;
        while (left < right) {
            int mid = left + (right - left) / 2;
            if (nums[mid] < target) {
                left = mid + 1;
            } else {
                right = mid;
            }
        }
```

```
        return nums[left] == target ? left : -1;
    }
}
```

二分查找右边界类似

这里大部分和寻找左边界是对称着来写的，唯独有一点需要尤其注意——中间位置的计算变了，我们在末尾多加了1。这样，无论对于奇数还是偶数，这个中间的位置都是偏右的。

```
class Solution {
    public int search(int[] nums, int target) {
        int left = 0;
        int right = nums.length - 1;
        while (left < right) {
            int mid = left + ((right - left) >> 1) + 1;
            if (nums[mid] > target) {
                right = mid - 1;
            } else {
                left = mid;
            }
        }
        return nums[right] == target ? right : -1;
    }
}
```

link