二分减治法:

普通二分模板：

While (left <= right):

Mid = left + (right – left) / 2

If nums[mid] == target:

Return mid

Elif nums[mid] < target:

Left = mid + 1

Else:

Right = mid – 1

这种写法可以看成是将待查找区间分为三个部分，[left, mid-1], [mid], [mid+1, right]。但是在出现多个相同元素的时候会有很多要考虑的因素，比如，要查找某元素的第一个出现的位置，当nums[mid]==target时，mid可能是解，也可能在左边还存在相同元素，所以应该走向哪一方有待进一步判断。

为了解决上述方法带来的不便，可以使用减治法的思想，将区间划分为包含目标元素的区间和不包含目标元素的区间，也就是说将区间划分为[left, mid][mid+1, right]或者[left, mid-1][mid, right]，具体要看题目划分。这样在每次更新区间时，只需要排除目标元素不在的那个区间，继续搜索包含目标元素的区间。当所有不满足的区间都排除掉后，最后剩下的就是满足的目标元素。

模板：

While (left < right):

Mid = left + (right – left) / 2

If (check(mid)):

# 这里要思考mid满足什么条件之后，应该如何排除不包含目标的区间

Right = mid – 1

Else:

Left = mid

# 在最后一定会有left==right，判断nums[left]是否满足条件，如果题目说一定存在解可以直接返回

Return left

需要注意的是，left=mid的赋值，由于更新mid是向下取整，可能存在只有两个元素从而导致死循环的情况，需要修改这种情况为向上取整。

参阅讲解地址：<https://leetcode-cn.com/problems/search-insert-position/solution/te-bie-hao-yong-de-er-fen-cha-fa-fa-mo-ban-python-/>