# 题目

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。

请必须使用时间复杂度为O(log n)的算法。

**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

提示:

1 <= nums.length <= 10^4

-10^4 <= nums[i] <= 10^4

nums 为无重复元素的升序排列数组

-10^4 <= target <= 10^4

# 分析

## 方法一：二分查找

思路：

本质上就是一个二分查找。

代码：

class Solution {

public:

int searchInsert(vector<int>& nums, int target) {

int n = nums.size();

int left = 0, right = n - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (nums[mid] > target) {

right = mid - 1;

} else if (nums[mid] < target) {

left = mid + 1;

} else {

return mid;

}

}

return left;

}

};

或：

class Solution {

public:

int searchInsert(vector<int>& nums, int target) {

if(nums.empty()) // 这个其实不需要

return 0;

int n = nums.size();

int left=0,right=n-1;

if(nums.at(left)>=target) // 这些过滤可以不加

return left;

if(nums.at(right)<target)

return right+1;

while(left<=right)

{

int mid = left+(right-left)/2;

if(nums.at(mid)<target)

left = mid+1;

else if(nums.at(mid)>target)

right=mid-1;

else

return mid;

}

return left;

}

};