Given a sorted array and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

You may assume no duplicates in the array.

**Example 1:**

**Input:** [1,3,5,6], 5

**Output:** 2

**Example 2:**

**Input:** [1,3,5,6], 2

**Output:** 1

**Example 3:**

**Input:** [1,3,5,6], 7

**Output:** 4

**Example 1:**

**Input:** [1,3,5,6], 0

**Output:** 0

题目：就是给定一个已经排好序的数列，然后要插入一个数字，如果已经存在则返回该元素的下标，如果不存在返回要插入的位置。

拿到这道题，我想到了二叉排序树的插入。但是似乎构造二叉排序树比较困难。这道题目仔细看没有涉及真的插入只是返回位置，所以二叉排序树并不好。

除了二叉排序树，应该想到的就是二分查找，**我们要想当二分查找，找不到的时候，low的位置正好是该数据应该所在的位置，high是该数据所在位置的上一位。**

例如：[1,3,5,6]，7

High low

[ 1 , 3 , 5 , 6 , ]

High最后值为3，low最后值为4.

[1,3,5,6], 0

High low

[ , 1 , 3 , 5 , 6 ]

High最后值为-1，low最后值为0.

当数据不存在的时候，我们可以看出low值最后所在的位置一定该数据插入时的所在位置。

代码：

public static int searchInsert(int[] nums, int target) {//典型的二分查找

int low = 0,high = nums.length-1;

int mid;

while(low<=high){

mid=(high+low)/2;

if(target==nums[mid])

return mid;

else if(target<nums[mid])

high=mid-1;

else

low=mid+1;

}

return low;

}

时间复杂度O()