二分查找

以下是一个简单的二分查找程序示例：

```
#include <iostream>
#include <vector>
#include <algorithm> // 用于 std::sort

// 二分查找函数
int binarySearch(const std::vector<int>& arr, int target) {
    int left = 0;
    int right = arr.size() - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2; // 防止(left + right)溢出

        if (arr[mid] == target) {
            return mid; // 找到目标，返回索引
        } else if (arr[mid] < target) {
            left = mid + 1; // 在右半部分查找
        } else {
            right = mid - 1; // 在左半部分查找
        }
    }

    return -1; // 未找到目标，返回-1
}

int main() {
    std::vector<int> numbers = {3, 6, 8, 12, 14, 17, 20, 25, 29, 31};
    int target = 17;

    // 确保数组是有序的
    std::sort(numbers.begin(), numbers.end());

    int result = binarySearch(numbers, target);

    if (result != -1) {
        std::cout << "Element found at index: " << result << std::endl;
    } else {
        std::cout << "Element not found in the array." << std::endl;
    }

    return 0;
}
```

代码说明

1. **输入数组**：`numbers`是一个有序的整数数组。在使用二分查找之前，确保数组是有序的。如果数组未排序，可以使用`std::sort`对其进行排序。
2. **二分查找实现**：
    - 初始化两个指针，`left`和`right`，分别指向数组的开始和结束位置。

- 计算中间位置`mid`。
- 如果`arr[mid]`等于目标值`target`，返回`mid`。
- 如果`arr[mid]`小于`target`，将`left`移动到`mid + 1`。
- 如果`arr[mid]`大于`target`，将`right`移动到`mid - 1`。
- 如果`left`超过`right`，表示目标值不在数组中，返回-1。

3. **输出结果**：根据`binarySearch`的返回值，输出目标值的位置或未找到的消息。

这个程序展示了如何在一个有序数组中使用二分查找来查找目标元素。二分查找的时间复杂度为O(log n)，对于大型数据集非常高效。

Leetcode经典例题：二分查找（35）

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。

请必须使用时间复杂度为`O(log n)`的算法。

示例 1:

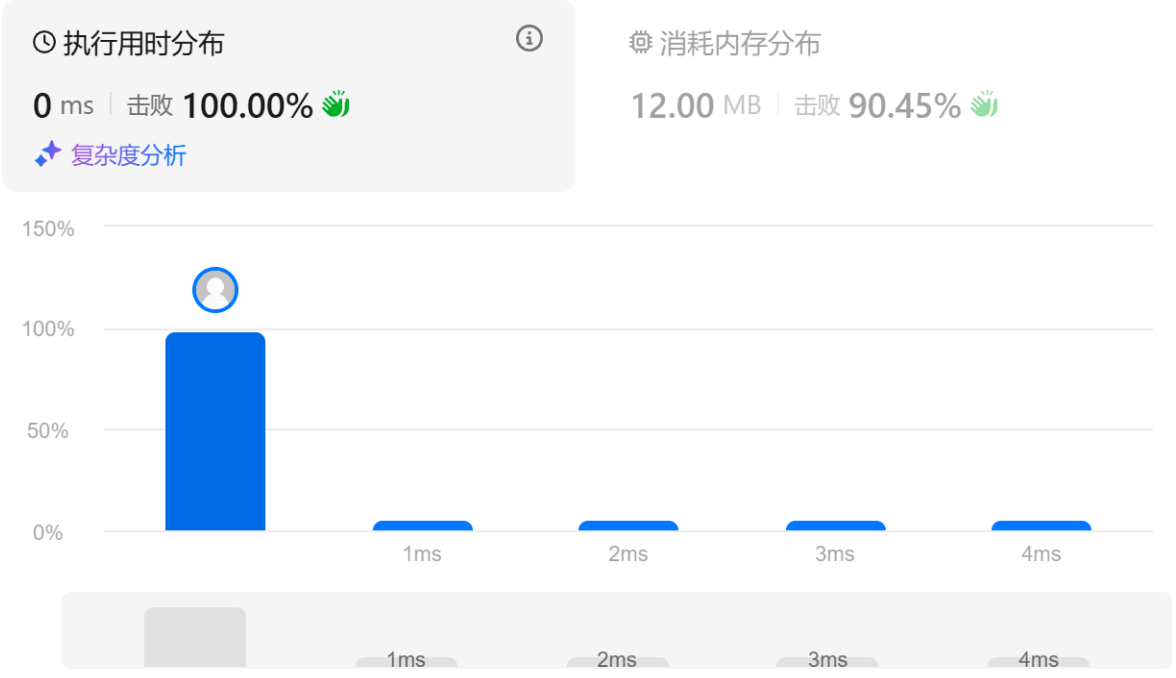
```
输入: nums = [1,3,5,6], target = 5
输出: 2
```

示例 2:

```
输入: nums = [1,3,5,6], target = 2
输出: 1
```

示例 3:

```
输入: nums = [1,3,5,6], target = 7
输出: 4
```



代码：

```
/*
 * @Author: Xiyuan Yang   xiyuan_yang@outlook.com
 * @Date: 2024-11-07 10:56:02
 * @LastEditors: Xiyuan Yang   xiyuan_yang@outlook.com
 * @LastEditTime: 2024-11-07 11:06:53
 * @FilePath: \CODE_for_Vscode\C_C++testcode\testcode20241107.cpp
 * @Description:
 * Do you code and make progress today?
 * Copyright (c) 2024 by Xiyuan Yang, All Rights Reserved.
 */
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
int searchInsert(vector<int>& nums, int target) {
        int i=0,j=nums.size()-1;
        int n=nums.size();
        int mid;
        while(i<j){
            mid=i+(j-i)/2;
            if(nums[mid]==target){
                return mid;
            }else if(nums[mid]>target){
                j=mid-1;
            }else{
                i=mid+1;
            }
        }
        mid=i;
        cout<<mid<<endl;
        if(nums[mid]==target){
            return mid;
        }else if(nums[mid]<target){
            return min(mid+1,n);
        }else{
            return mid;
        }
    }
int main(){
    vector <int> nums={1,3};
    cout<<searchInsert(nums,2);
}
```