# 题目

给定一个已按照升序排列的有序数组，找到两个数使得它们相加之和等于目标数。

函数应该返回这两个下标值index1和index2，其中index1必须小于index2。

**说明:**

返回的下标值（index1和index2）不是从零开始的。

你可以假设每个输入只对应唯一的答案，而且你不可以重复使用相同的元素。

**示例:**

输入: numbers = [2, 7, 11, 15], target = 9

输出: [1,2]

解释: 2与7之和等于目标数9。因此index1 = 1, index2 = 2。

# 分析

## 方法一：双指针

class Solution {

public:

vector<int> twoSum(vector<int>& numbers, int target) {

int low = 0,high = numbers.size()-1;

vector<int> ret;

while(low < high)

{

int sum = numbers[low] + numbers[high];

if(sum == target)

{

ret.push\_back(low+1);

ret.push\_back(high+1);

return ret;

}

else if(sum < target)

{

low++;

}

else

{

high--;

}

}

return ret;

}

};

或：

class Solution {

public:

vector<int> twoSum(vector<int> &numbers, int target) {

int left = 0, right = numbers.size() - 1;

while (true) {

int s = numbers[left] + numbers[right];

if (s == target) return {left + 1, right + 1}; // 题目要求下标从 1 开始

s > target ? --right : ++left;

}

}

};

复杂度分析：

时间复杂度：O(n)，其中n为numbers的长度。

空间复杂度：O(1)，仅用到若干变量。

## 方法二：哈希表