# 题目

给你一个整数数组 citations ，其中 citations[i] 表示研究者的第 i 篇论文被引用的次数。计算并返回该研究者的 h 指数。

根据维基百科上 h 指数的定义：h 代表“高引用次数” ，一名科研人员的 h 指数 是指他（她）至少发表了 h 篇论文，并且 至少 有 h 篇论文被引用次数大于等于 h 。如果 h 有多种可能的值，h 指数 是其中最大的那个。

示例 1：

输入：citations = [3,0,6,1,5]

输出：3

解释：给定数组表示研究者总共有 5 篇论文，每篇论文相应的被引用了 3, 0, 6, 1, 5 次。

由于研究者有 3 篇论文每篇至少被引用了3次，其余两篇论文每篇被引用不多于3次，所以她的h指数是 3。

示例 2：

输入：citations = [1,3,1]

输出：1

提示：

n == citations.length

1 <= n <= 5000

0 <= citations[i] <= 1000

# 分析

## 方法一：排序

首先我们可以将初始的 H指数 h 设为0，然后将引用次数排序，并且对排序后的数组从大到小遍历。

根据 H指数的定义，如果当前 H指数为 h并且在遍历过程中找到当前值 citations[i]>h，则说明我们找到了一篇被引用了至少 h+1次的论文，所以将现有的 h值加1。继续遍历直到 h无法继续增大。最后返回 h作为最终答案。

代码：

class Solution {

public:

int hIndex(vector<int>& citations) {

sort(citations.begin(), citations.end());

int h = 0, i = citations.size() - 1;

while (i >= 0 && citations[i] > h) {

h++;

i--;

}

return h;

}

};

复杂度分析

时间复杂度：O(nlogn)，其中n为数组citations的长度。即为排序的时间复杂度。

空间复杂度：O(logn)，其中n为数组 citations的长度。即为排序的空间复杂度。

## 方法二：计数排序