题目：

Given an array of citations (each citation is a non-negative integer) of a researcher, write a function to compute the researcher's h-index.

According to the definition of h-index on Wikipedia: "A scientist has index *h* if *h* of his/her *N* papers have **at least** *h* citations each, and the other *N − h* papers have **no more than** *h* citations each."

For example, given `citations = [3,0,6,1,5]`, which means the researcher has `5` papers in total and each of them had received `3, 0, 6, 1, 5` citations respectively. Since the researcher has `3` papers with **at least** `3` citations each and the remaining two with **no more than** `3` citations each, his h-index is `3`.

Note: If there are several possible values for `h`, the maximum one is taken as the h-index.

1.时间：O（NLOGN）; 空间：O（1）

```
class Solution {
public:
    int hIndex(vector<int>& citations) {
        const int size = citations.size();
        int result = 0;
        std::sort(citations.begin(), citations.end());
        for (int i = 0; i < size; ++i){
            int tmp = citations[i];
            if (size - i <= citations[i]){
                return (size-i);
            }
        }

        return 0;
    }
};
```

2.时间：O（NLOGN）; 空间：O（1） ->比1慢

```
class Solution {
public:
    int hIndex(vector<int>& citations) {
        const int size = citations.size();
        int result = 0;
        std::sort(citations.begin(), citations.end());
        for (int i = 1; i <= size; ++i){
            if (citations[size - i] >= i){
                result = i;
            } else{
                break;
            }
        }
        return result;
    }
};
```

3.时间：O（N）; 空间：O（N）   ->计数排序变种

```
class Solution {
public:
    int hIndex(vector<int>& citations) {
        const int size = citations.size();
        std::vector<int> hashTable(size + 1, 0);
        /* 统计计数 */
        for (int i = 0; i < size; ++i){
            if (citations[i] >= size){
                hashTable[size]++;
            }
            else{
                hashTable[citations[i]]++;
            }
        }

        /*for (int i = size - 1; i >= 0; --i){
            hashTable[i] = hashTable[i] + hashTable[i + 1];
            }
            int result = 0;
            for (int i = size - 1; i >= 0; --i){
            if (hashTable[i] >= i) return hashTable[i];
            }*/
        if(hashTable[size] >= size) return size;

        for (int i = size - 1; i >= 0; --i){
            hashTable[i] += hashTable[i + 1];
            if (hashTable[i] >= i) return i;
        }

        return 0;
    }
};
```