1 给出一个 Java 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."

Example:
Input: citations = [3,0,6,1,5]
Output: 3

Explanation:
* [3,0,6,1,5] 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, her h-index is 3.
* Note: If there are several possible values for h, the maximum one is taken as the h-index.

2 解决思路、编写 Java 代码、执行

思路：先把数组从大到小排序，再从头遍历，找到符合要求的 h-index 值。

[3,0,6,1,5]➔[6,5,3,1,0]

遍历时，如果数组中该位置的数值大于等于其序号+1，则 h-index 至少为序号+1。

例如：

- 第 0 位为 6，6>=1，则 h-index=1；
- 第 1 位为 5，5>=2，则 h-index=2；
- 第 2 位为 3，3>=3，则 h-index=3；
- 第 3 位为 1，1<4，则 h-index 不再变化，算法终止。

在 Eclipse 中构造工程，命名为 SC2020Spring_Classroom_Exercise；
在 src 中创建包，命名为 exercise_6_6；
在包中创建类 HIndex.java。

(1) 从控制台读入用户输入，按 3,0,6,1,5 的格式，存储于数组

```
    Scanner scanner = new Scanner(System.in);
    int[] citations = new int[100];
    String[] strs;
    System.out.println("Please input the citation numbers:");
    String line = scanner.nextLine();
    strs = line.split(",");
    for (int i = 0; i < strs.length; i++)
        citations[i] = Integer.parseInt(strs[i]);
```

(2) 编写排序功能（冒泡排序）

```
    for (int i = 0; i < number - 1; i++) {
        for (int j = 0; j < number - 1; j++) {
            if (citations[j] < citations[j + 1]) {
                int temp = citations[j + 1];
                citations[j + 1] = citations[j];
                citations[j] = temp;
            }
        }
    }
```

(3) 计算 h-index

```
    int hindex = 0;
    for (int j = 0; j < number; j++) {
        if (citations[j] >= j + 1)
            hindex = j + 1;
        else
            break;
    }
    System.out.println("The h-index is: " + hindex);
```

输入不同的数组，进行手工测试。
上述编写完成后，git commit 到 v0.1 All code in main()

(4) 提取出单独的 int hindex(int[] citations)函数
(5) 提取出来形成单独的排序函数 void sort(int[] array)

git commit 到 v0.2 separate functions

(6) 从文本文件读入，存储于数组（学生自己完成）

3 健壮性处理

进行健壮性测试：

* 输入空数组——抛出异常了

```
String line = new String();
line = scanner.nextLine();
while(line.length() == 0) {
    System.out.println("Input empty, please re-input:");
    line = scanner.nextLine();
}
```

git commit 到 v0.3 avoid empty input string

* 输入负值：学生自己完成

策略：split 之后，调用 Integer.parseInt(strs[i])得到整数，检查其是否<0

* 输入特殊值（非整数、非法字符等）

策略 1：比较笨的方法：逐个字符位置检查是否为数字（学生自己实现）

策略 2：与负值一起处理，用正则表达式检查分割后字符串是否匹配"[0-9]+"

策略 3：直接调用 Integer.*parseInt*()或者 Integer.*valueOf*().intValue()，若抛出异常，说明不合法，需要在捕获异常之后让用户重新输入

这里实现策略 2：

```
//read input from keyboard
Scanner scanner = new Scanner(System.in);
System.out.println("Please input the citation numbers:");

int[] citations = new int[100];
String[] strs;
String line = new String();

//loop, until user inputs legal string
while (true) {
    line = scanner.nextLine();
    //if the input is empty
    if (line.length() == 0) {
        System.out.println("Input empty, please re-input:");
        continue;
    }
```

```
    //check if each part is integer >= 0
    boolean legalNumbers = true;
    strs = line.split(",");
    for (int i = 0; i < strs.length; i++) {
        //if not, stop checking others and let user re-input
        if(! strs[i].matches("[0-9]+")) {
            System.out.println(strs[i] + " is illegal: ");
            legalNumbers = false;
            break;
        }
        //otherwise, store the integer into array
        citations[i] = Integer.parseInt(strs[i]);
    }
    if (!legalNumbers)
        continue;
    else {
        //calculate h-index
        int hindex = hindex(citations);
        //output to console
        System.out.println("The h-index is: " + hindex);
        break;
    }
}
```

git commit 到 v0.4 avoid input string containing illegal characters

* 超过 100 个输入怎么办?——抛出异常了

不能提前得知用户输入多少个

策略：使用 Java Collections, List, Set, Map, etc

```
//int[] citations = new int[5];
List<Integer> citations = new ArrayList<>();

//citations[i] = value;
citations.add(value);

int hindex = hindex(citations);

public static int hindex(List<Integer> citations) {
    //Integer[] array1 = (Integer[])citations.toArray();
    int[] array2 = new int[citations.size()];
    for(int i=0; i<citations.size(); i++)
        array2[i] = citations.get(i);
```

```
    // 冒泡排序
    sort(array2);

    // 计算h-index
    int hindex = 0;
    for (int j = 0; j < array2.length; j++) {
        if (array2[j] >= j + 1)
            hindex = j + 1;
        else
            break;
    }
    return hindex;
}
```

git commit 到 v0.5 use java collections instead of arrays

4 将计算功能与用户输入分离开来

从 main 中分离出来计算，main 只处理调用（作为客户端程序）

```
public class HIndex {

    private List<Integer> citations = new ArrayList<>();

    public HIndex3(String input) {
        if(input == null || input.length() == 0)
            throw new IllegalArgumentException("Empty");

        dealInput(input);
    }

    private void dealInput(String input) {

        String[] strs = input.split(",");

        for (int i = 0; i < strs.length; i++) {
            if(! strs[i].matches("[0-9]+"))
                throw new IllegalArgumentException
                                (strs[i] + " is illegal");

            citations.add(Integer.parseInt(strs[i]));
        }
    }
```

```
    public static void main(String[] args) {
        String[] inputs = new String[] {"1,0", "3,-2,4,8"};
        for(int i=0;i<inputs.length;i++) {
            HIndex3 h = new HIndex3(inputs[i]);
            System.out.println(h.calcHIndex());
        }
    }
    public int calcHIndex() {...}
    ...
}
```

git commit 到 v0.6 separate input with calculation

通过静态函数进行调用?

```
        for(int i=0;i<inputs.length;i++) {
            //HIndex3 h = new HIndex3(inputs[i]);
            System.out.println(HIndex3.calcHIndex(inputs[i]));
        }
```

git commit 到 v0.7 use static methods of a class

5 直接编写测试用例

打开 Junit，演示测试用例如何书写、如何运行、正确和错误的执行结果

创建一个新的测试类 HIndexTest.java
演示如何编写测试函数并启动执行、查看结果
正确的、错误的

普通的用例如何测试：assertEquals
抛出异常如何测试？
    只看抛出异常的类型是否正确
    还要看抛出异常的消息是否正确

最好每个测试方法里只写一个测试用例

git commit 到 v0.8 design test cases manually

6 根据等价类和边界值思想，设计和编写测试用例

设计更完备的测试用例

Testing strategy：
    对输入的字符串进行等价类划分

针对每一个划分结果来设计测试用例并撰写测试方法

git commit 到 v0.9 design test cases by equivalence partitioning

学生可以自己补充更完备的测试用例

7 改造为 OOP

构造 Paper 类，存储论文的题目、年份、期刊、引用数，封装起来
对 Paper 类进行功能扩展，包括增加引用数、减少引用数；

基于 Paper 类改造之前的 HIndex 类，将其改造为 Author 类，其中管理作者名字、发表论文清单，使用集合类表达一组 Paper 而不再用数组。

在 Author 类中增加功能：新增论文、修改某个论文的引用数、计算 HIndex、toString（第一行打印出作者，后面每行是一篇论文[题目、引用数]，最后是该作者的 HIndex）。

写一个客户端（Author 类的 main 函数，模拟使用 Paper 和 Author 类）

git commit 到 v1.0 oop

让 Paper 类具备在 Collections 中的排序功能：
Collections.sort(xxx);

方案 0：自行编写排序功能
方案 1：使用 Comparator
方案 2：使用 Comparable

实现方案 2
git commit 到 v1.1 use comparator for sorting papers

其他方案学生可自己实现

8 重新编写和执行 JUnit 测试用例

对 Paper 类的各函数进行测试
对 Author 类的各函数进行测试

学生请自行实现

9 构造 GUI

为用户开发一个 GUI，让用户输入一组论文题目和引用数，计算出 HIndex
使用 JFrame，创建新 GUI 类

学生请自行实现

10 读取文本文件

在 GUI 上选择一个文件，从文本文件里读取数据，写入 Paper 对象和 Collections。

学生请自行实现