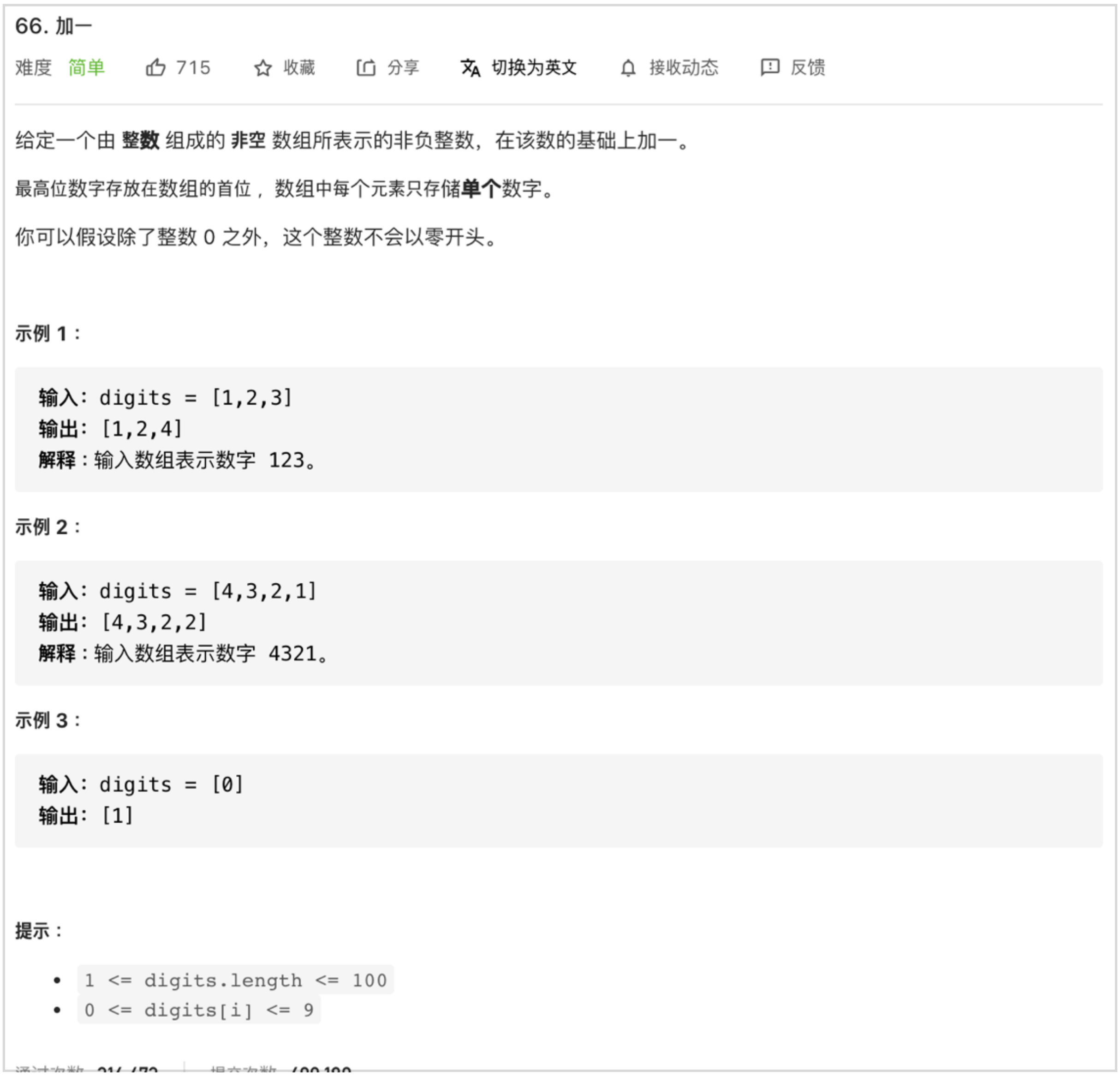
66. 加一

地址

加一

题目



66. 加一

难度 简单 715 收藏 分享 切换为英文 接收动态 反馈

给定一个由 **整数** 组成的 **非空** 数组所表示的非负整数，在该数的基础上加一。

最高位数字存放在数组的首位， 数组中每个元素只存储**单个**数字。

你可以假设除了整数 0 之外，这个整数不会以零开头。

示例 1：

```
输入：digits = [1,2,3]
输出：[1,2,4]
解释：输入数组表示数字 123。
```

示例 2：

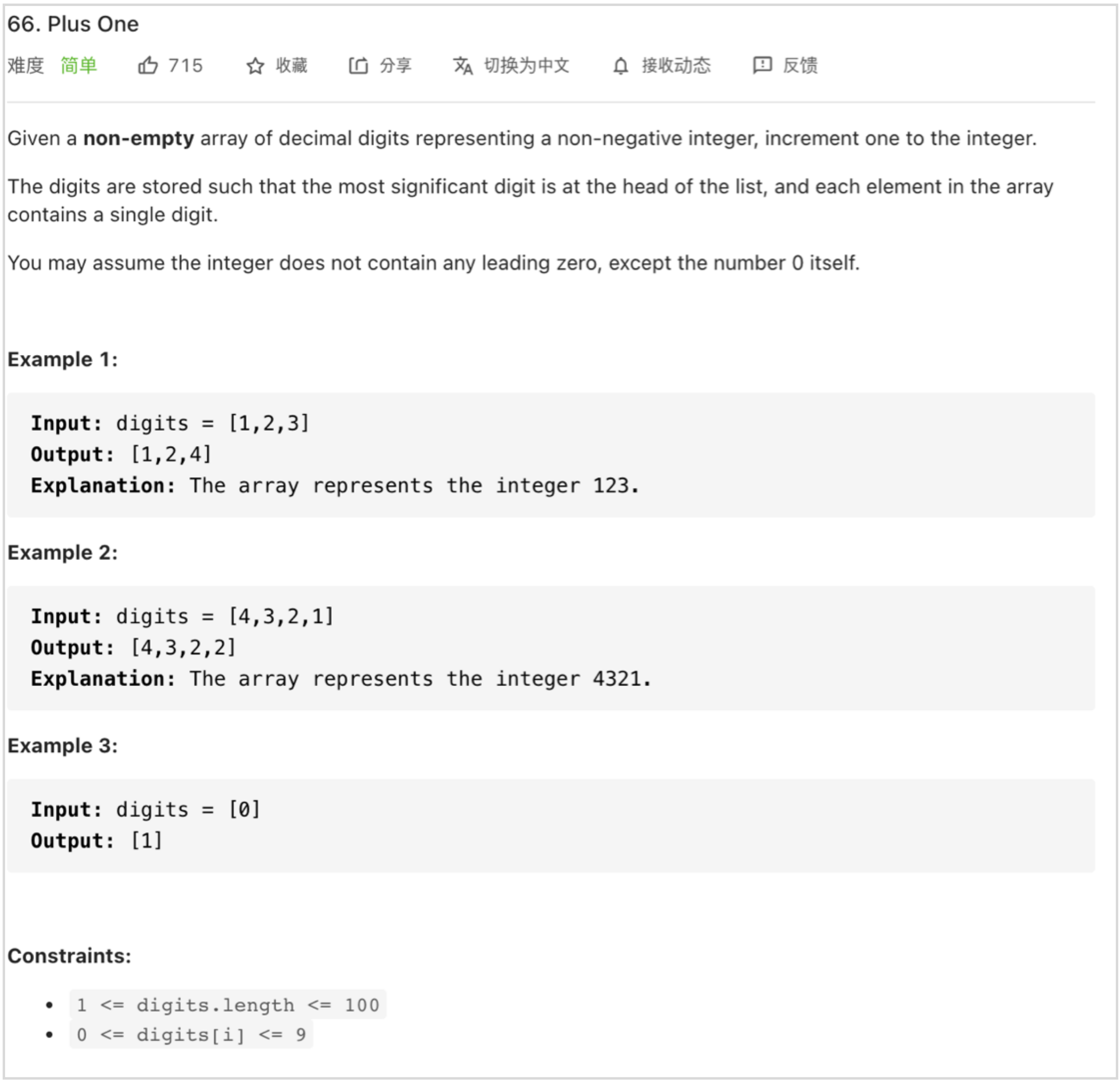
```
输入：digits = [4,3,2,1]
输出：[4,3,2,2]
解释：输入数组表示数字 4321。
```

示例 3：

```
输入：digits = [0]
输出：[1]
```

提示：

- `1 <= digits.length <= 100`
- `0 <= digits[i] <= 9`



66. Plus One

Given a **non-empty** array of decimal digits representing a non-negative integer, increment one to the integer.

The digits are stored such that the most significant digit is at the head of the list, and each element in the array contains a single digit.

You may assume the integer does not contain any leading zero, except the number 0 itself.

Example 1:

```
Input: digits = [1,2,3]
Output: [1,2,4]
Explanation: The array represents the integer 123.
```

Example 2:

```
Input: digits = [4,3,2,1]
Output: [4,3,2,2]
Explanation: The array represents the integer 4321.
```

Example 3:

```
Input: digits = [0]
Output: [1]
```

Constraints:

- `1 <= digits.length <= 100`
- `0 <= digits[i] <= 9`

思路 1 ： 暴力解法

复杂度分分析

- 时间复杂度：O(n^2)
- 空间复杂度：O(1)

代码

```
        // Java
        // Time : 2021 - 07 -18

    public int[] plusOne(int[] digits) {
```

```java
        digits[digits.length - 1]++;

        int addition = digits[digits.length - 1];

        int carry = addition / 10;

        int remainder = addition % 10;

        if (carry == 0) { // no carry
            return digits;
        }

        if (digits.length == 1) {
            digits = new int[2];
            digits[0] = carry;
            digits[1] = remainder;
            return digits;
        }



        for (int i = digits.length - 2; i >= 0; i--) {
            digits[i+1] = remainder;
            digits[i] += carry;
            addition = digits[i];
            carry = addition / 10;
            remainder = addition % 10;

            if (carry == 0) {
                return digits;
            }

            digits[i] = remainder;

            if (i == 0) {

                if (carry == 0) {
                    return digits;
                }
                digits[i] = remainder;
                int[] res = new int[digits.length + 1];
                res[0] = carry;

                for (int j = 0, k = 1; j < digits.length && k < res.length; j+
+, k++) {
                    res[k] = digits[j];
                }

                digits = new int[digits.length + 1];
                digits = res;
            }
        }

        return digits;
    }
```

#Leetcode/Array#