66. Plus One

Description | Hints | Submissions | Discuss | Solution

Pick One

Given a **non-empty** array of digits representing a non-negative integer, plus 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 contain a single digit.

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

Example 1:

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

Example 2:

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

```java
public class L66 {
      //对数组进行反转
       public int [] arrayInverse(int [] a) {
           int [] b = new int [a.length];
           for(int i = 0; i < a.length; i ++) {
               b[a.length - i - 1] = a[i];
           }
           return b;
       }

       public int[] plusOne(int[] digits) {
            if(digits == null)
                return null;

            int [] temp = new int [digits.length];

            digits = new L66().arrayInverse(digits);
            int i = 0;
            //判断是否需要进位，如果为10，则需要进位，如果不为，则直接跳出
            for( ; i < digits.length; i++) {
                temp[i] = digits[i] + 1;
                  if(temp[i] != 10)
                      break;
                  else {
                      temp[i] = 0;
                }
            }

            for(int j = i+1; j < digits.length; j++){
                temp[j] = digits[j];
            }

            int [] plus = null;
            //这个看是否最后一位需要进位。
            if(i == digits.length) {
                plus = new int[digits.length + 1];
                int j = 0;
                for( ; j < digits.length - 1 ; j++) {
                      plus[j] = temp[j];
                }
                plus[j] = 0;
                plus[j + 1] = 1;
            }else {
                plus = new int[digits.length];
                for(int j = 0 ; j < digits.length; j++) {
                      plus[j] = temp[j];
                }
            }
            //返回是反转后的结果
            return new L66().arrayInverse(plus);
        }


}
```