135. Candy

Hard 400 105 Favorite Share

There are *N* children standing in a line. Each child is assigned a rating value.

You are giving candies to these children subjected to the following requirements:

- Each child must have at least one candy.
- Children with a higher rating get more candies than their neighbors.

What is the minimum candies you must give?

Example 1:

```
Input: [1,0,2]
Output: 5
Explanation: You can allocate to the first, second and third child with 2,
1, 2 candies respectively.
```

Example 2:

```
Input: [1,2,2]
Output: 4
Explanation: You can allocate to the first, second and third child with 1,
2, 1 candies respectively.
             The third child gets 1 candy because it satisfies the above two
conditions.
```

Accepted 95,878 | Submissions 347,493

```java
package Algorithm;

import java.util.Arrays;

/*
 * 1、与前面的邻居比较，前向遍历权重数组ratings，如果ratings[i]>ratings[i-1]，则A[i]=A[i-1]+1;
2、与后面的邻居比较，后向遍历权重数组ratings，如果ratings[i]>ratings[i+1]且A[i]<A[i+1]+1，则更新A，A[i]=A[i+1]+1;
3、对A求和即为最少需要的糖果。
 */
public class L135 {
    public int candy(int[] ratings) {
        int [] A = new int [ratings.length];
        //最少为1
        Arrays.fill(A, 1);

        for(int i = 1; i < A.length; i ++) {
            if(ratings[i] > ratings[i - 1])
                A[i] = A[i - 1] + 1;
        }

        int sum = A[A.length - 1];
        //这里是从后面开始，为什么要从后面再来一遍，因为如果开始权重是1 2 8 3 2 1
        //经过前面的比较后，会变成1 2 3 1 1 1 1 ，但是第四个权重为3，比后面的要重，但是只是1
        for(int i = A.length - 2; i >= 0; i --) {
            if(ratings[i] > ratings[i + 1] && A[i] <= A[i + 1])
                A[i] = A[i + 1] + 1;
            sum += A[i];
        }
        return sum;
    }
}

```