209. **Minimum Size Subarray Sum**

Difficulty Medium 715 47 Discuss Favorite Share

Description | Hints | Solution | Submissions

Given an array of **n** positive integers and a positive integer **s**, find the minimal length of a **contiguous** subarray of which the sum ≥ **s**. If there isn't one, return 0 instead.

Example:

```
Input: s = 7, nums = [2,3,1,2,4,3]
Output: 2
Explanation: the subarray [4,3] has the minimal length under the problem constrain
```

Follow up:
If you have figured out the *O*(*n*) solution, try coding another solution of which the time complexity is *O*(*n* log *n*).

```java
public class L209 {

    /*
     * 这个其实是动态规划的问题，对每一个数组元素计算从它到之前的之和大于s的最小长度
     * 然后看是不是最小的长度，如果是则替换，如果不是，则往前。
     */
    public int minSubArrayLen(int s, int[] nums) {
        int sum = 0;
        int val = 0;
        if(nums == null || nums.length == 0)
            return sum;

        int i = 0;
        int j = 0;
        while (i <= j && j < nums.length) {
            boolean flag = false;
              val += nums[j];
              while (val >= s) {
                  flag = true;
                  val -= nums[i];
                  i ++;
              }

              if((sum == 0 || sum > j - i + 1) && flag) {
                  sum = j - i + 2;
              }
              j ++;
       }

        return sum;
    }

    public static void main(String [] args) {
        int [] nums = new int [] {2,3,1,2,4,3};
        System.out.println(new L209().minSubArrayLen(7, nums));
    }
}
```