JAVA-数组滑窗-有 N 个正整数组成的一个序列
目描述：
有 N 个正整数组成的一个序列。给定整数 sum，求长度最长的连续子序列，使他们的和等于 sum，返回此子序列的长度，如果没有满足要求的序列，返回-1。

输入描述：
序列：1,2,3,4,2

sum：6
输出描述：
序列长度：3

补充说明：
输入序列仅由数字和英文逗号构成，数字之间采用英文逗号分隔；
序列长度：1 <= N <= 200；
输入序列不考虑异常情况，由题目保证输入序列满足要求。
例 1
输入：
1,2,3,4,2
6
输出：
3
说明：
解释：1,2,3 和 4,2 两个序列均能满足要求，所以最长的连续序列为 1,2,3，因此结果为 3
示例 2
输入：
1,2,3,4,2
20
输出：
-1
说明：
解释：没有满足要求的子序列，返回-1

```
import java.util.Scanner;

// 注意类名必须为 Main, 不要有任何 package xxx 信息
public class Main {
        public static void main(String[] args) {
//              int[] arr = {
////                          1, 2, 3, 4, 2, 1, 1, 1, 5, 6, 8, 4, 2, 2, 3, 4, 5, 6, 1, 2, 3, 5, 7, 8, 9, 6, 3, 3, 1, 1, 5,
6, 6, 3, 6, 6, 9, 8, 4
////                          1, 2, 3, 4, 2, 1, 1, 1
////                          1, 2, 3, 4, 2
//                          1, 2, 3, 4, 2, -3
```

```
//            };
//            System.out.println(getLongestSub(arr, -1));
//            System.out.println(getLongestSub2(arr, -1));
//            System.out.println(getLongestSub(arr, 35));
//            System.out.println(getLongestSub2(arr, 35));
//            System.out.println(getLongestSub(arr, 10));
//            System.out.println(getLongestSub2(arr, 10));
//            System.out.println(getLongestSub(arr, 3));
//            System.out.println(getLongestSub2(arr, 3));
//            System.out.println(getLongestSub(arr, 5));
//            System.out.println(getLongestSub2(arr, 5));
//            System.out.println(getLongestSub(arr, 20));
//            System.out.println(getLongestSub2(arr, 20));

        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        int sum = sc.nextInt();
        String[] split = s.split(",");
        int[] arr = new int[split.length];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = Integer.parseInt(split[i]);
        }
        System.out.println(getLongestSub(arr, sum));
    }

    private static int getLongestSub2(int[] arr, int sum) {
        int res = -1;
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j <= i; j++) {
                int t = 0;
                for (int k = j; k <= i; k++) {
                    t += arr[k];
                }
                if (t == sum) {
                    res = Math.max(res, i - j + 1);
                }
            }
        }
        return res;
    }

    private static int getLongestSub(int[] arr, int sum) {
        if (arr == null || arr.length == 0) {
            return -1;
```

```
        }
        int res = -1;
        int[] sums = new int[arr.length];
        sums[0] = arr[0];
        for (int i = 1; i < arr.length; i++) {
            sums[i] = sums[i - 1] + arr[i];
        }
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j <= i; j++) {
                int t = 0;
                for (int k = j; k <= i; k++) {
                    t += arr[k];
                }
                if (t == sum) {
                    res = Math.max(res, i - j + 1);
                }
            }
        }
        return res;
    }
}
```