去除多余空格

知识点 字符串数组 队列
时间限制：2s 空间限制：256MB 限定语言：不限

题目描述：

去除文本多余空格，但不去除配对单引号之间的多余空格。给出关键词的起始和结束下标，去除多余空格后刷新关键词的起始和结束下标。

输入：
Life is painting a picture, not doing 'a sum'.
8 15,20 26,43 45

输出：
Life is painting a picture, not doing 'a sum'.
[8, 15][19, 25][42, 44]

条件约束：

1，不考虑关键词起始和结束位置为空格的场景；

2，单词的的开始和结束下标保证涵盖一个完整的单词，即一个坐标对开始和结束下标之间不会有多余的空格；

3，如果有单引号，则用例保证单引号成对出现；

4，关键词可能会重复；

5，文本字符长度length取值范围：[0, 100000]；

输入描述：

输入为两行字符串：
第一行：待去除多余空格的文本，用例保证如果有单引号，则单引号成对出现，且单引号可能有多对。
第二行：关键词的开始和结束坐标，关键词间以逗号区分，关键词内的开始和结束位置以单空格区分。

例如：

Life is painting a picture, not doing 'a sum'.
8 15,20 26,43 45
关键单词为：painting picture sum

输出描述：

输出为两行字符串：
第一行：去除多余空格后的文本
第二行：去除多余空格后的关键词的坐标开始和结束位置，为数组方式输出。

例如：

Life is painting a picture, not doing 'a sum'.
[8, 15][19, 25][42, 44]

示例1

输入：

Life is painting a picture, not doing 'a sum'.
8 15,20 26,43 45

输出：

Life is painting a picture, not doing 'a sum'.
[8, 15][19, 25][42, 44]

说明：

a和picture中间多余的空格进行删除

示例2

输入：

Life is painting a picture, not doing 'a sum'.
8 15,19 25,42 44

输出：

Life is painting a picture, not doing 'a sum'.
[8, 15][19, 25][42, 44]

说明：

a和sum之间有多余的空格，但是因为有成对单引号，不去除多余空格

解题思路：

例如：

Life is painting a picture, not doing 'a sum'.
8 15,20 26,43 45
第一个字符[8,15]是painting

1. 先判断是否在括号内：左侧单引号为偶数0，说明不在括号内
2. 左侧多余括号为0
3. 右侧多余括号为0

结论：坐标位置不变[8,15]
第二个字符[20,26]是picture

1. 先判断是否在括号内：左侧单引号为偶数0，说明不在括号内
2. 左侧多余括号为1
3. 右侧多余括号为0

结论：坐标整体向前移动1，坐标为[19,25]
第三个字符[43,45]是sum

1. 先判断是否在括号内：左侧单引号为奇数1，说明在括号内

结论：字符串在括号内，坐标不变。但是前面的字符总共有1个空格，所有坐标需要向前移动1，坐标为[42,44]

```
public class Main{

    public static String inputStr;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        inputStr = sc.nextLine();
        String[] strings = sc.nextLine().split(",");

        List<int[]> zuobiaoList = new ArrayList<>();
        int count = 0;    //空格个数
        String resStr = inputStr;

        for(String str : strings){

            String[] strs = str.split(" ");

            int start = Integer.parseInt(strs[0]);    //起始坐标
```

```
                int end = Integer.parseInt(strs[1]);       //末尾坐标
                int spaceLeft = 0;       //字符串左侧空格数
                int spaceRight = 0;         //字符串右侧空格数
                int[] ints = new int[2];
                if(!isInKuohao(start)){         //在括号内不处理
                    spaceLeft = spaceLeft(start);
                    spaceRight = spaceRight(end);
                }

                ints[0] = start - count - spaceLeft;       //起始坐标-之前去除的空格总数-当前左侧的空格数
                ints[1] = end - count - spaceLeft;          //末尾坐标-之前去除的空格总数-当前左侧的空格数

                while (resStr.charAt(ints[0]) == ' '){          //起始坐标为空，则进行删除（去除多余的空格）
                    resStr = resStr.substring(0, ints[0]) + resStr.substring(ints[0] + 1);
                }

                count += spaceLeft + spaceRight;       //记录总共去除的空格数
                zuobiaoList.add(ints);
            }

            System.out.println(resStr);
            String res = "";
            for(int[] ints : zuobiaoList){
                res += "[" + ints[0] + "," + ints[1] + "]";
            }

            System.out.println(res);
        }

        /**
         * 字符串左侧多余空格数
         * @param index
         * @return
         */
        public static int spaceLeft(int index){

            int count = 0;

            if(index == 0){
                return count;
            }else {
```

```
            while (true){
                index --;
                if(index >= 0 && inputStr.charAt(index) == ' '){
                    count ++;
                }else {
                    break;
                }
            }
        }

        return count > 1 ? count - 1 : 0;     //空格数小于等于 1 时，多余空格数为 0
}

/**
 * 字符串右侧的多余空格数
 * @param index
 * @return
 */
public static int spaceRight(int index){

        int count = 0;

        if(index == 0){
            return count;
        }else {
            while (true){
                index ++;
                if(index < inputStr.length() && inputStr.charAt(index) == ' '){
                    count ++;
                }else {
                    break;
                }
            }
        }

        return count > 1 ? count - 1 : 0;     //空格数小于等于 1 时，多余空格数为 0
}

/**
 * 是否在括号内（字符左侧的单引号为单数则在括号内，双数则在括号外）
 * @param start         字符串其实坐标
 * @return
 */
public static boolean isInKuohao( int start){
```

```
            String str1 = inputStr.substring( 0, start);

            int count = 0;
            for(int i=0; i<str1.length(); i++){
                  if(str1.charAt(i) == '\''){
                        count ++;
                  }
            }

            return count%2 == 0 ? false : true;
      }

}
```