

JAVA-跳房子 I

题目描述：

跳房子，也叫跳飞机，是一种世界性的儿童游戏。

游戏参与者需要分多个回合按顺序跳到第 1 格直到房子的最后一格。跳房子的过程中，可以向前跳，也可以向后跳。

假设房子的总格数是 `count`，小红每回合可能连续跳的步数都放在数组 `steps` 中，请问数组中是否有一种步数的组合，可以让小红两个回合跳到最后一格？如果有，请输出索引和最小的步数组合。

注意：数组中的步数可以重复，但数组中的元素不能重复使用。提供的数据保证存在满足题目要求的组合，且索引和最小的步数组合是唯一的。

输入描述：

第一行输入为每回合可能连续跳的步数，它是 `int` 整数数组类型。实际字符串中整数与逗号间可能存在空格。

第二行输入为房子总格数 `count`，它是 `int` 整数类型。

输出描述：

返回索引和最小的满足要求的步数组合（顺序保持 `steps` 中原有顺序）

补充说明：

`count` <= 1000，`0` <= `steps.length` <= 5000，`-100000000` <= `steps[i]` <= `100000000`

示例 1

输入：

[1,4,5,2,2]

7

输出：

[5, 2]

说明：

示例 2

输入：

[-1,2,4,9,6]

8

输出：

[-1, 9]

说明：

此样例有多种组合满足两回合跳到最后，譬如: [-1,9]，[2,6]，其中[-1,9]的索引和为  $0+3=3$ ，[2,6]的索引和为  $1+4=5$ ，所以索引和最小的步数组合[-1,9]

```
import java.util.*;
```

```
// 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) {  
        Scanner in = new Scanner(System.in);  
        String inStr = in.nextLine();  
        int count = in.nextInt();  
        inStr = inStr.substring(1, inStr.length() - 1);  
        String[] str = inStr.split(",");  
        int[] steps = new int[str.length];  
        for (int i = 0; i < steps.length; i++) {  
            steps[i] = Integer.parseInt(str[i].trim());  
        }  
        if(steps.length<2){  
            return;  
        }  
        int minIndex = Integer.MAX_VALUE;  
        int m1 =0;  
        int m2 =0;  
        List<Integer> list = new ArrayList<>();  
        for(int i=0;i<steps.length-1;i++){  
            for(int j=i+1;j<steps.length;j++){  
                if(steps[i]+steps[j]==count&& i+j<minIndex){  
                    minIndex = i+j;  
                    m1=i;  
                    m2=j;  
                }  
            }  
        }  
        list.add(steps[m1]);  
        list.add(steps[m2]);  
        System.out.println(list);  
    }  
}
```