题目描述：给定一个小写字母组成的字符串s，请找出字符串中两个不同位置的字符作为分割点，使得字符串分成的三个连续子串且子串权重相等，注意子串不包含分割点。
若能找到满足条件的两个分割点，请输出这两个分割点在字符串中的位置下标，若不能找到满足条件的分割点请返回0,0。
子串权重计算方式为：子串所有字符的ASCII码数值之和。

输入描述：输入为一个字符串，字符串由a~z，26个小写字符组成，5 <= 字符串长度 <= 200。

输出描述：输出为两个分割点在字符串中的位置下标，以逗号分隔

补充说明：只考虑唯一解，不存在一个输入多种输出解的情况

示例1
输入：acdbbbca
输出：2,5
说明：以位置2和5作为分割点，将字符串分割为ac，bb，ca三个子串，每一个的子串权重都为196，输出为：2,5

示例2
输入：abcabc
输出：0,0
说明：找不到符合条件的分割点，输出为0,0

```
const rl = require("readline").createInterface({ input: process.stdin });
var iter = rl[Symbol.asyncIterator]();
const readline = async () => (await iter.next()).value;

void async function () {
    let str
    while(line = await readline()){
        let tokens = line.split(' ');
        str=tokens[0]
    }
    function find(str){
        let n=str.length;
        if(n<5){return [0,0]}
        let weight= new Array(n).fill(0);
        weight[0]=str[0].charCodeAt(0);
        for(let i=1;i<n;i++){
            weight[i]=weight[i-1]+str[i].charCodeAt(0);
        }
        for(let i=1;i<n-3;i++){
            for(let j=i+2;j<n-1;j++){
                const left =weight[i-1];
                const mid = weight[j-1]-weight[i];
                const right =weight[n-1]-weight[j];
                if(left == mid&& mid==right){
                    return [i,j]
                }
            }

        }
        return [0,0];
    }
```

```
    let m = find(str);
    console.log(m.join(","))

}()
```