喊 7 的次数重排题目描述：

喊 7 是一个传统的聚会游戏，N 个人围成一圈，按顺时针从 1 到 N 编号。编号为 1 的人从 1 开始喊数，下一个人喊的数字为上一个人的数字加 1，但是当将要喊出来的数字是 7 的倍数或者数字本身含有 7 的话，不能把这个数字直接喊出来，而是要喊"过"。假定玩这个游戏的 N 个人都没有失误地在正确的时机喊了"过"，当喊到数字 K 时，可以统计每个人喊"过"的次数。

现给定一个长度为 N 的数组，存储了打乱顺序的每个人喊"过"的次数，请把它还原成正确的顺序，即数组的第 i 个元素存储编号 i 的人喊"过"的次数。

输入描述：

输入为一行，为空格分隔的喊"过"的次数，注意 K 并不提供，K 不超过 200，而数字的个数即为 N。

输出描述：

输出为一行，为顺序正确的喊"过"的次数，也由空格分隔。

补充说明：

示例 1

输入：

```
0 1 0
```

输出：

```
1 0 0
```

说明：

一共只有一次喊"过"，那只会发生在需要喊 7 时，按顺序，编号为 1 的人会遇到 7，故输出 100。注意，结束时的 K 不一定是 7，也可以是 8、9 等，喊过的次数都是 100。

示例 2

输入：

```
0 0 0 2 1
```

输出：

```
0 2 0 1 0
```

说明：

一共有三次喊"过"，发生在 7 14 17，按顺序，编号为 2 的人会遇到 7 17，编号为 4 的人会遇到 14，故输出 02010。

```
const rl = require("readline").createInterface({ input: process.stdin });

var iter = rl[Symbol.asyncIterator]();

const readline = async () => (await iter.next()).value;


void async function () {

    // Write your code here

    while(line = await readline()){

        const initArr = line.split(' ').map(Number)

        const skipArr = initArr.filter(item=>item!=0)

        const skipNum = eval(skipArr.join('+'))

        const num = initArr.length

        let arr = new Array(num).fill(0)

        let i = 1
```

```
        let tempNum = 0
        while(tempNum<skipNum){
            const flag = i%7 == 0 || String(i).includes('7')
            if(flag){
                const yushu = i%num
                if(yushu > 0) {
                    arr[yushu-1] += 1
                } else if(yushu == 0) {
                    arr[arr.length -1] +=1
                }
                tempNum++
            }
            i++
        }
        console.log(arr.join(' '))
    }
}()
```