字符串统计

题目描述：

给定两个字符集合，一个为全量字符集，一个为已占用字符集。已占用的字符集中的字符不能再使用，要求输出剩余可用字符集。

输入描述：

1、输入为一个字符串，一定包含@符号。@前的为全量字符集，@后的字为已占用字符集。

2、已占用字符集中的字符一定是全量字符集中的字符。字符集中的字符跟字符之间使用英文逗号分隔。

3、每个字符都表示为字符加数字的形式，用英文冒号分隔，比如 a:1，表示 1 个 a 字符。

4、字符只考虑英文字母，区分大小写，数字只考虑正整形，数量不超过 100。

5、如果一个字符都没被占用，@标识仍然存在，例如 a:3,b:5,c:2@

输出描述：

输出可用字符集，不同的输出字符集之间回车换行。

注意，输出的字符顺序要跟输入一致。不能输出 b:3,a:2,c:2

如果某个字符已全被占用，不需要再输出。

示例 1

输入：

```
a:3,b:5,c:2@a:1,b:2
```

输出：

```
a:2,b:3,c:2
```

说明：

全量字符集为 3 个 a，5 个 b，2 个 c。
已占用字符集为 1 个 a，2 个 b。
由于已占用字符不能再使用，因此，剩余可用字符为 2 个 a，3 个 b，2 个 c。
因此输出 a:2,b:3,c:2

```
package main

import (
	"bufio"
	"fmt"
	"os"
	"strconv"
	"strings"
)

func main() {
	reader := bufio.NewReader(os.Stdin)
```

```
    line, _, _ := reader.ReadLine()
    sets := strings.Split(string(line), "@")
    all := strings.Split(sets[0], ",")
    use := strings.Split(sets[1], ",")
    var allSequeue string
    allMap := make(map[string]int)
    for i := 0; i < len(all); i++ {
        charData := strings.Split(all[i], ":")
        c := charData[0]
        allSequeue += c
        cCount, _ := strconv.Atoi(charData[1])
        allMap[c] = cCount
    }
    for i := 0; i < len(use); i++ {
        charData := strings.Split(use[i], ":")
        if len(charData) != 2 {
            continue
        }
        c := charData[0]
        cCount, _ := strconv.Atoi(charData[1])
        allMap[c] -= cCount
    }
    var res []string
    for i := 0; i < len(allSequeue); i++ {
        if c := string([]byte{allSequeue[i]}); allMap[c] > 0 {
            res = append(res, fmt.Sprintf("%s:%d", c, allMap[c]))
        }
    }
    fmt.Print(strings.Join(res, ","))
}
```