2023 研发组一面后端考核

> In software architecture, there may be many layers between the hardware and end user.The *back* usually handles data storage and business logic.
>
> 在软件架构中，硬件和最终用户之间可能有很多层。而后端通常处理数据存储和业务逻辑。

以上是维基百科在软件架构中对于后端的解释。如果你不太明白，那也没有关系，因为你已经怀揣着一颗炽热且热爱的心来到这里，我们相信你能够应对接下来的挑战。现在你即将开始运用 Golang 编写你的第一个后端 Demo 程序，迈出打破常规的第一步。

任务描述

常见的应用场景，手机号+验证码的登陆验证。

简化版操作需要实现功能：

1. 在手机号正确和验证码有效的情况下，显示登录成功
2. 在输入手机号后
    a. 输入 1： 表示使用验证码登录
    b. 输入 2： 表示获取验证码
    c. 输入其他： 表示未知操作
3. 其余情况下显示登录失败，并显示失败信息

例如：

```
请输入电话号码
13535814223
1:输入验证码进行登录 2:获取验证码
1
请输入验证码登录
22213
无效验证码
1:输入验证码进行登录 2:获取验证码
2
验证码为： 058684
1:输入验证码进行登录 2:获取验证码
2
一分钟内已获取验证码无法重新获取
1:输入验证码进行登录 2:获取验证码
1
请输入验证码登录
058684
登录成功
```

需要注意的点：

1. 检验输入的手机号格式
2. 验证码是随机 6 位 0-9 的数字
3. 60 秒内无法重新发送验证码
4. 验证码的有效期是 5 分钟，在验证码的有效期内都可以使用验证码进行成功登录

> 可能会用到的包：time，rand，regexp..... 可自行使用搜索引擎进行学习相关知识

任务要求

1. 独立完成，可以使用搜索引擎
2. 使用 Golang 实现
3. 能够输出正确的提示信息

一些加分项（不作强制要求）

1. 可以尽量多地使用函数、结构体、方法等，而不是将所有逻辑全部堆积在 main 函数中，或许在这个任务中的实现来说完全没有必要，但这是个展示所学知识的一个好方法。
2. 良好的代码规范

3. 限制单个手机号当日最多发送 5 次验证码
4. 实现一个更复杂的验证码生成策略，例如包含字母和数字的验证码
5. 尝试对验证码进行加密后再存储，并保证系统的基本功能不变。
6. 尝试对数据进行持久化
7. 了解并使用 Golang 的 Gofmt，对代码进行格式化

注意事项

1. 请不要对任何人透露本文档，包括但不限于截图、 复制、转发、口述等。
2. 本次考核任务没有标准答案，你们可以选用自己喜欢或熟悉的方式来进行实现。对于输出的提示信息能正确表达目的即可，不需要与样例完全一致。对于未规定的部分，可以自由发挥，保证程序合理即可。
3. 在完成任务期间，你可以使用搜索引擎，也仍然可以向学长学姐询问与任务无关的问题，但请不要无脑复制粘贴，相对于技术，我们更看重真诚。
4. 只需尽力完成即可，无法完美实现并不代表什么，将最完美的自己展现给我们就 Enough 了。
5. 请在 9 月 21 日 23:59 前将你的代码文件发送至邮箱:develop@ncuhome.club（注意备注好个人信息）或是上传至你的 GitHub 仓库（注意将你的 GitHub 账号名告知我们），以便我们提前查看你的代码，也为我们更加充分地准备面试打下基础。