从零基础开始的考核项目学习文档

项目实现过程：

Level0-1；

Git的使用仍然不清楚，只是跟着教程走，磕磕绊绊能够上传。

Level0-2；

在b站上略看了一遍翁恺的c语言基础教程，并试着敲了几行代码。

然后在知乎上搜索了gcc的使用，简要理解了编译的过程。

Level1-1；

1-1比较简单，但在写的过程中渐渐熟悉了各语句的语法规则（特别是可恶的双等号判断符）。

也进行了一些较深入的思考，改进了第一个版本，定义了一个电梯移动的函数，便于实现后面的level。

Level1-2；

在网上学习了简单的文件读入语句，实现了文件输入。

（其他代码与level1-1几乎完全一致）

Level1-3；

思维的难度上来了。

首先是策略的选择问题，参考了知乎回答后，我选择了LOOK算法。

这也是日常生活中的单电梯运行策略。

但是对于一些特殊的情况，例如：

电梯满载，放下一个人后处于下行状态，而电梯的上一层有人要上电梯

这时可能的最佳策略是先上去接人。

但是由于临界条件不好确定，

且从心理学的角度，乘客不希望电梯忽上忽下，

故不采用这种策略。

然后是代码实现的问题：

1、首次处理了较复杂的逻辑。

2、通过测试发现了逻辑的不严密处。

3、初次使用结构体数组。

4、思考了代码效率的问题。

Level2-1；

在1-3的基础上，我在结构体中添加了时间T这个参数，

并将它考虑到决策中来，整个过程还算比较顺利。

在完成了2-2之后，我根据2-2的思路重构了2-1，变成了一个新的版本。

（2-1理论上支持同一楼层有多个呼叫的情况）

Level2-2；

负载均衡策略：

一开始想着将之前的代码物尽其用，于是想出了负载均衡策略。

将所有的请求分为两部分，由两部电梯分别按照2-1的方式去完成。

虽然很取巧，但是这个策略至少会比一部电梯效率更高。

实时决定策略：

想了很久，有很多情况要讨论，而且比较复杂。

最后综合考虑心理学，概率学以及生活中电梯实际的运行情况，

写出了一个差不多接近于生活中电梯实际情况的策略。

（理论上支持同一楼层有多个呼叫的情况）

代码实现：

1. 改变了之前的程序构架，用时间参数进行循环。
2. 增加了电梯结构体数组。
3. 使用了很多条件判断语句和for循环。
4. 在不断测试的过程中发现问题，不断改正。

总的收获：

1. 在干中学，遇到不会的技术问题通过一些渠道寻找帮助，

在完成这些项目的过程中我有很大的提升。

1. 全心投入在项目上，不断遇到问题，自己不断去解决，

是一次很好的锻炼。

1. 不断地去反思和优化，优化算法、程序构架、代码效率。

不定期地去回顾之前写的代码，能够找到一些能够优化的地方。

认识到的不足：

1. 自己从零开始，不知道的东西还有很多。
2. 代码量不够，不知道该怎么去更合理地构建代码。

提高代码效率。同时使用了一些goto语句，使得代码的可读性变低。

1. 算法思维以及思维逻辑性还有待加强。

总之，20天来收获颇多，非常享受这样的过程。

非常希望加入Dian团队，在干中学，不断提升。