初始化：收到dian团队的招新题目后，我按提示在runoob上对git进行了学习。从在官网下载安装git，在cmd上进行更改用户名和邮箱地址，我改为了qdzhang和我自己的邮箱。并用init指令对仓库初始化，但不太理解暂存区以及与之相关的add和mv等指令在实际工作中的作用。同时进行了VScode的安装和dev的调整。

0-1：我在github网站上注册了账号，并建立了名为qdzhang1的私人远程仓库和qdzhang2的公共远程仓库（url在0-1.txt中），最后按指示建立了ssh密匙并建立了ssh形式的远程仓库qdzhang3(ssh.jpg,ssh2.jpg)，并上传了一个README文件作为测试（测试.jpg）。但我没能完整记住命令的字母拼写和其中-+字母参数的含义。

0-2：因为之前有过信竞的经历，我又一次阅读了《啊哈C》等书籍，复习了以前在提高组学习的基本语法，输入输出，if，函数，数组，文件输入输出等知识。

1-1：这里我的理解是只有一位乘客在一层楼上，是无需算法的模拟，直接对电梯运行方式进行模拟。输出格式与题干中相同。

1-2：这里我使用freopen重定向的方式，对1-1中的代码稍加改动，输入文件为dian.in，输出文件为dian.out，同时因为题干中出现了字符串“elevator”，我复习了关于字符串读取的内容，以%s为参数读入。

1-3：刚开始看到题面时，我想到了用深度优先搜索在每次接到乘客或送下乘客时对向上向下两个方向进行枚举，时间复杂度为2的2n次方，在本题中约为10的7次方，可在0.1s内完成。但算法普适性不强，在楼层数过大（大于20）或人数多时容易消耗时间过长。在阅读电梯调度算法后，由于没有单个目标的时限和优先级，按下按钮时间也相同，因此只能在SSTF（最短寻找楼时间优先）和扫描的LOOK（感觉LOOK明显比单纯的SCAN更快，而且也省电）。联系我自己的生活经历，我感觉电梯运营方更加希望不会出现少量特殊个体等待过长时间（方差过大）从而导致投诉，因此我选用LOOK算法，同时加入对超载的判断使其能够在超载时寻找同方向上的下客点。输入输出格式如下：

输入文件为dian2.in，输出文件为dian2.out

输入：

第一个行包含两个正整数N和M，表示有乘客的楼层数量（即乘客数量）与电梯所在楼层数

接下来N行，每行包含a，b两个正整数，表示这位乘客在a层按动电梯按钮，目的地为b层

输出：

不定行数，每行分为三个数字c，d，e

表⽰在c层的时候，电梯运⾏d个单位时间，有e⼈

2-1：由于没有时限的限制，基本算法和1-3相似，输入输出格式如下：

输入文件为dian3.in，输出文件为dian3.out

输入：

正整数M，N，分别表示电梯此时所在的楼层与乘客总数量（数据总量）

接下来N行，每行a,b,t三个正整数，表示距离上一组数据t时后（初始为0），在a层有乘客按下电梯按钮，目的地为b层。

输出：

不定行数，每行分为三个数字c，d，e

表⽰在c层的时候，电梯运⾏d个单位时间，有e⼈

2-2：基本与2-1一致。新任务出现时：两电梯均空闲时选最近的；一个空闲时如所需方向与另一个不同则选用空闲的电梯，否则选用有任务的电梯；均繁忙时则优先方向相同，其次距离更近。输入输出格式如下：

输入文件为dian4.in，输出文件为dian4.out

输入：

正整数M1，M2，N，分别表示1电梯此时所在的楼层、2电梯此时所在的楼层与乘客总数量（数据总量）

接下来N行，每行a,b,t三个正整数，表示距离上一组数据t时后（初始为0），在a层有乘客按下电梯按钮，目的地为b层。

输出：

分为两部分，第一部分：

第一行为“电梯1：“

不定行数，每行分为三个数字c1，d1，e1

表⽰在c1层的时候，1电梯运⾏d1个单位时间，有e1⼈

第二部分：

第一行为“电梯2：“

不定行数，每行分为三个数字c2，d2，e2

表⽰在c2层的时候，2电梯运⾏d2个单位时间，有e2⼈