00 第五次作业说明文档

一、输入要求

可以采用一行输入多个指令，或一行输入一个指令的模式输入。最终输入一行单独的大写 END 后程序结束。

楼层为一位或两位正整数，数字前最多允许二十个前导零，最多允许数字前出现一个正号或负号。

楼层共二十层，最底层只有向上运行请求按钮，最顶层只有向下运行请求按钮，中间楼层同时有向上和向下两个方向请求按钮。程序运行开始或重置时设置电梯停靠在一层。（请求省略要求同指导书，此不赘述）

从主请求发出时刻起（包括发出时刻）到主请求到达目标楼层开门（不包括开门时刻）止，对满足可捎带的请求进行捎带。

捎带方法见指导书。

电梯不能突然改变运动状态

从 main 函数开始执行开始，记为开始时间 0。

若存在“未完成”的捎带请求，则在主请求后将第一个未完成捎带请求变为主请求。

楼层请求格式为：(FR, m, UP/DOWN)，其中 FR 为标识，m 为发出请求的楼层号，UP 为向上请求，DOWN 为向下请求

电梯内请求格式为：(ER, #e，n，其中 ER 为标识，e 为请求电梯，n 为请求去往的目标楼层号

括号内的逗号应采用 ASCII 字符集中的逗号“,”，而不是中文字符逗号“，”。请求之间可以使用空格、Tab 和换行（不可使用分号！出现即视为非法字符）分隔（额外提示，如第二次作业公测样例中括号间出现逗号的情况，程序会报错并提取正确部分，即报错并输出正确结果）。请求内部元素之间可以有空格。请求之中允许使用空格、Tab，但不允许使用换行分隔。

当一行输入存在错误时，若其中存在正确指令，仍选取其中正确的指令保留，无论是否有保留，弹出错误提示（此在后文中详细说明）。

二、输出要求

每个不被忽略的正确指令操作会产生一行输出。

忽略请求会出现 SAME[指令]

按捎带顺序输出，优先输出捎带指令，再输出 SAME 指令，再输出主请求。

由于使用 printf 式调试方法，所以在程序中出现很多的中间状态，如：

```
----------- Start Work -----------
(FR,5,UP)
Elevator1 1->5
Elevator1 MOVING(3s)
Elevator1 FINISH MOVING
Elevator1 2->5
Elevator1 MOVING(3s)
Elevator1 FINISH MOVING
Elevator1 3->5
Elevator1 MOVING(3s)
Elevator1 FINISH MOVING
Elevator1 4->5
Elevator1 MOVING(3s)
Elevator1 FINISH MOVING
(1491408204071):[FR,5,UP,9.0]/(#1,5,UP,4,21.0)
Elevator1 OPEN THE DOOR
Elevator1 CLOSE THE DOOR
```

其中红色部分为线程 sleep 状态，如果此时输入正好与下一条输出相撞，有可能出现死机（等已有指令跑完之后可以继续输入），建议避免措施是快速的输入，或者可以将这些额

外输出注释掉

具体注释方法：（可能后期更改行数会改变，所以截图表示具体的代码内容）

Elevator 类

第 141、144、146 行

```
//move a floor
System.out.println("Elevator"+ctrl+" "+floornow+"->"+floororder);
try{
    //Modify the moving +3s
    System.out.println("Elevator"+ctrl+" MOVING(3s)");
    sleep(3000);
    System.out.println("Elevator"+ctrl+" FINISH MOVING");
}catch(InterruptedException e){
```

第 198、200 行

```
if(judge_shun && fi!=floororder){//if by the way in the way
    try{//modify open the door
        System.out.println("Elevator"+ctrl+" OPEN THE DOOR(6s)");
        sleep(6000);
        System.out.println("Elevator"+ctrl+" FINISH OPEN THE DOOR");
        time+=6.0;
    }catch(InterruptedException e){

    }
```

第 218、220 行

```
if(!s.equals("STILL")){
    try{//modify open the door //for the main_request
        System.out.println("Elevator"+ctrl+" OPEN THE DOOR(6s)");
        sleep(6000);
        System.out.println("Elevator"+ctrl+" FINISH OPEN THE DOOR");
        time+=6.0;
    }catch(InterruptedException e){
    }
}
```

第 383 行与 385 行

```
try{
    System.out.println("Elevator"+ctrl+" OPEN THE DOOR");
    sleep(6000);
    System.out.println("Elevator"+ctrl+" CLOSE THE DOOR");
}catch(InterruptedException e){
```

注释后结果大致如下：

```
------------ Start Work ------------
(FR,5,UP)
(1491408696065):[FR,5,UP,11.8]/(#1,5,UP,4,23.8)
```

感觉红色 sleep 状态更长更难以琢磨……还是建议不注释为好，如果只是想看最后输出可以不看控制台，看 result.txt 文件即可

当键入单独一行大写 END 时，程序控制台会提示你结束，但实际线程可能未结束，所有线程结束后输出结束语，注意，当输入完 END 后，END 之后再输入的指令被忽略

大致结果：

```
---------- Start Work ----------
(FR,3,UP)
Elevator1 1->3
Elevator1 MOVING(3s)
END
(FR,3,UP)(FR,3,UP)(FR,3,UP)
Elevator1 FINISH MOVING
Elevator1 2->3
Elevator1 MOVING(3s)
Elevator1 FINISH MOVING
(1491410829864):[FR,3,UP,3.0]/(#1,3,UP,2,9.0)
Elevator1 OPEN THE DOOR
Elevator1 CLOSE THE DOOR

------------ End Work ------------
```

注意，欢迎语和中间字符不会出现在 result.txt 文件中

三、容错（以下报错均为输入一行键入回车后立即输出）报错前均有系统时间

当楼层为 20 楼，但申请 UP 指令时，报错“Illegal Input!(The house has a roof!)”

当楼层为 1 楼，但申请 DOWN 指令时，报错“Illegal Input!(There's no basement!)”

当请求指令的楼层数不为 1~20 及之间的正整数时，报错“Illegal Input!(Where are you going?”

当存在其他不符合标准的输入时，报错“Illegal Input!”

当请求的电梯号不为 1~3 间正整数时，报错“Illegal Input!(Invalid Elevator

Number!)”

当一行出现超过 10 条有效指令（same 指令仍然算有效指令，无效指令仅包括出现错误输入的指令，而不考虑开关门反复按键等问题）时，报错“Illegal Input!(More than 10 right requests in same line!)”

其余未知错误，如输入 ctrl+Z 等，报错“Something was wrong!”

当指令被忽略时，输出 SAME[request]

四、其他细节

在电梯调度时，同等运动量时，优先安排号码较小的电梯去工作，即优先度 Elevator1 > Elevator2 > Elevator3