面向对象设计与构造第二单元第二次作业指导书

第一部分：训练目标

本次作业的基本目标是模拟**多线程实时电梯系统**，在第一次作业的基础上，掌握线程安全知识并解决线程安全问题，同时在架构上围绕线程之间的协同设计层次架构。

第二部分：基本概念

- 1、电梯系统时间 T_{real}：从程序开始运行，到所有线程终止，程序结束的时刻花费的时间，即程序的真实运行时间 (real time)
- 2、电梯系统运行花费总时间 T_{final}：从程序开始运行，到电梯最后输出关门的时刻花费的时间
- 3、开关门窗口时间：从电梯开始开门的时刻到完成关门的时刻的时间闭区间
- 4、数据基本限制：所有测试数据均需满足的限制
- 5、公测数据限制：中测和强测数据满足的限制
- 6、互测数据限制：互测数据需满足的限制

第三部分：题目描述

一、电梯系统说明

本次作业需要模拟一个多线程环形实时电梯系统。

系统基于一个类似北京航空航天大学新主楼的大楼，大楼有 A, B, C, D, E 五个座，每个楼座可以对应多台电梯，可以在楼座内 $1-10$ 层之间运行。**本次作业将新增楼座与楼座之间移动的环形电梯**，每个楼层可以对应多台环形电梯，可以在 $A \to B \to C \to D \to E \to A$ 或者 $A \to E \to D \to C \to B \to A$ 所组成的回路之间运行，例如，位于A 座 2 楼的横向电梯可以直接运行到 B 座 2 楼 / E 座 2 楼。

系统从标准输入中输入请求信息，程序进行接收和处理，模拟电梯运行，将必要的运行信息通过输出接口进行输出。

具体而言，本次作业电梯系统具有的功能为：

- 1、楼座对应的电梯：上下行，开关门，以及模拟乘客的进出。
- 2、楼层对应的电梯：左右行，开关门，以及模拟乘客的进出。

电梯系统**可以采用任意的调度策略**，即在任意时刻，系统选择上下（左右）行动，是否在某层开关门，都可自定义，只要保证在**电梯系统时间不超过系统时间上限**的前提下将所有的乘客送至目的地即可。

电梯每纵向（横向）运行一层（座）、开关门的时间为固定值，仅在**开关门窗口时间内允许乘客进出。**

电梯系统默认初始在 A, B, C, D, E 五个楼座的1层中各有一个纵向电梯。

二、电梯参数说明

- 1、可到达楼层：1-10层
- 2、纵向电梯初始位置：各座1层
- 3、横向电梯初始位置：A 座各层
- 4、数量：最初5部纵向电梯，可增加

- 5、编号：最初5部电梯中 A 座为1号，B 座为2号，依次类推；新增的电梯编号自定义，但不可与之前的重复
- 6、纵向电梯移动一层花费的时间：0.4s
- 7、横向电梯移动一座花费的时间：0.2s
- 8、开门花费的时间：0.2s
- 9、关门花费的时间：0.2s
- 10、限乘人数：6人

三、电梯请求说明

本次作业的电梯，为一种比较特殊的电梯，学名叫做**目的选择电梯。**

大概意思是，**在电梯的每个入口，都有一个输入装置，让每个乘客输入自己的目的位置。**电梯基于这样的一个目的地选择系统进行调度，将乘客运送到指定的目标位置。

所以，一个电梯请求包含这个人的**出发楼座、出发楼层、目的楼座和目的楼层**，以及这个人的 id（保证人员 id 唯一），请求内容将作为一个整体送入电梯系统，在整个运行过程中请求内容不会发生改变。

四、电梯捎带规则说明

在**不违反课程规则**的前提下，我们对电梯的**捎带策略不做限制。**希望同学们多多阅读并自行探索相关算法，对电梯的捎带策略有自己的设计与思考。

为保证同学们实现的都为作业要求的**可捎带电梯**，我们对电梯的性能做一定的约束，采用**类似ALS的调度策略**为性能**基准**（关于调度策略，欢迎大家积极上网找资料探索）。性能约束详见公测说明-正确性判断章节中关于 T_{max} 的部分。

五、基准策略

- 1、对于纵向电梯，采用 ALS 策略
- 2、对于横向电梯，采用类似 ALS 的策略
  - (1)、**目标方向**为能够完成这个请求的最短路线所走的方向，比如 A 座到 C 座采用 $A \rightarrow B \rightarrow C$ 的策略，A 座到 E 座采用 $A \rightarrow E$ 的策略
  - (2)、分为**主请求**和**被捎带请求**两个概念
  - (3)、主请求选择规则：
    - a、如果电梯中没有乘客，将请求队列中到达时间最早的请求作为主请求
    - b、如果电梯中有乘客，将其中到达时间最早的乘客请求作为主请求
  - (4)、被捎带请求选择规则：
    - a、电梯的主请求存在
    - b、该请求投喂的时刻**小于等于**电梯到达该请求出发楼座关门的截止时间
    - c、电梯主请求的目标方向和该请求的目标方向一致
- 3、在电梯的调度上，采用一种较为均衡的调度方式，例如 A 座 7 层有5个乘客，3部电梯，那么第1个乘客分配给第1部电梯，第2个乘客分配给第2部电梯，第3个乘客分配给第3部电梯，第4个乘客分配给第1部电梯，第5个乘客分配给第2部电梯

第四部分：输入/输出和样例

一、输入输出接口说明

- 1、本单元作业使用官方提供的输入输出接口进行输入输出。
- 2、输入接口提供了若干方法，通过调用方法可以直接获取所需数据对象。输入接口在**尝试**读取标准输入的内容时，对于正确的输入将成功解析，错误的则输出错误信息，但不会影响程序的正常运行。
- 3、输出子程序接受一个字符串，并附上时间戳后再输出。
- 4、详细的接口定义和使用方法见官方接口说明文档。
- 5、程序的输入输出为**实时交互**，评测机可以做到在某个时间点投放一定量的输入。

二、输入数据

每行输入数据可分以下三类：

- 1、请求信息：
  - (1)、格式为：`[时间戳]乘客ID-FROM-起点座-起点层-TO-终点座-终点层`
  - (2)、输入保证 $[\text{起点座} == \text{终点座}] + [\text{起点层} == \text{终点层}] == 1$，**即楼座和楼层两者中有且仅有一个相同**，换言之，乘客一次只能在同一层楼或者同一座楼移动，从 A 座 1 层到 C 座 3 层是不被允许的。
  - (3)、乘客 ID 唯一，且满足ID是 int 类型数据
- 2、为一个楼座动态增加纵向电梯的指令：
  - (1)、格式为：`ADD-building-电梯ID-楼座ID`
  - (2)、输入保证楼座 $ID \in [A, B, C, D, E]$
  - (3)、电梯 ID 唯一
  - (4)、新建电梯默认处于对应楼座 1 层
- 3、为一个楼层动态增加横向电梯的指令：
  - (1)、格式为：`ADD-floor-电梯ID-楼层ID`
  - (2)、输入保证楼层 $ID \in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$
  - (3)、电梯 ID 唯一
  - (4)、新建电梯默认处于对应楼层 A 座

三、输出数据

- 1、通过调用输出接口的每一次输出将自动附加时间戳于头部，具体请参考**输出接口文档**，请不要试图伪造时间戳，否则会在实际评测时产生不一样的结果导致程序运行错误。
- 2、电梯到达某一位置：`[时间戳]ARRIVE-所在座-所在层-电梯ID`

  > **每到达一个楼层都要输出，初始所在的楼层不需要输出**

- 3、电梯**开始**开门：`[时间戳]OPEN-所在座-所在层-电梯ID`
- 4、电梯**完成**关门：`[时间戳]CLOSE-所在座-所在层-电梯ID`
- 5、乘客进入电梯：`[时间戳]IN-乘客ID-所在座-所在层-电梯ID`
- 6、乘客离开电梯：`[时间戳]OUT-乘客ID-所在座-所在层-电梯ID`

四、样例

#	输入	输出	说明
1	[0.0]ADD-building-6-A [1.0]2-FROM-A-5-TO-A-3 [2.3]3-FROM-A-2-TO-A-9	[1.2940]ARRIVE-A-2-6 [1.7150]ARRIVE-A-3-6 [2.1180]ARRIVE-A-4-6 [2.5370]ARRIVE-A-5-6 [2.5380]OPEN-A-5-6 [2.5920]ARRIVE-A-2-1 [2.5940]OPEN-A-2-1 [2.7520]IN-2-A-5-6 [2.8020]IN-3-A-2-1 [2.9720]CLOSE-A-5-6 [3.0230]CLOSE-A-2-1 [3.3920]ARRIVE-A-4-6 [3.4400]ARRIVE-A-3-1 [3.7970]ARRIVE-A-3-6 [3.7990]OPEN-A-3-6 [3.8520]ARRIVE-A-4-1 [4.0150]OUT-2-A-3-6 [4.2240]CLOSE-A-3-6 [4.2740]ARRIVE-A-5-1 [4.6830]ARRIVE-A-6-1 [5.0960]ARRIVE-A-7-1 [5.5130]ARRIVE-A-8-1 [5.9290]ARRIVE-A-9-1 [5.9320]OPEN-A-9-1 [6.1550]OUT-3-A-9-1 [6.3680]CLOSE-A-9-1	根据标准的调度策略，2号请求由6号电梯完成，3号请求由1号电梯完成
2	[1.0]ADD-floor-10-7 [2.1]1-FROM-C-7-TO-A-7	[2.3230]ARRIVE-B-7-10 [2.5400]ARRIVE-C-7-10 [2.5410]OPEN-C-7-10 [2.7560]IN-1-C-7-10 [2.9730]CLOSE-C-7-10 [3.1890]ARRIVE-B-7-10 [3.3060]ARRIVE-A-7-10 [3.3080]OPEN-A-7-10 [3.5230]OUT-1-A-7-10 [3.7390]CLOSE-A-7-10	对于初次增加电梯的楼层，确保在 1s 后才会出现对该楼层的请求

3	[0.0]1-FROM-A-8-TO-A-7 [0.8]ADD-floor-7-2 [4.3]2-FROM-B-2-TO-C-2 [7.9]ADD-floor-8-6 [9.1]3-FROM-D-6-TO-A-6 [9.2]ADD-floor-9-1 [9.3]4-FROM-B-1-TO-D-1	[0.4380]ARRIVE-A-2-1 [0.8410]ARRIVE-A-3-1 [1.2510]ARRIVE-A-4-1 [1.6550]ARRIVE-A-5-1 [2.0560]ARRIVE-A-6-1 [2.4610]ARRIVE-A-7-1 [2.8670]ARRIVE-A-8-1 [2.8690]OPEN-A-8-1 [3.0850]IN-1-A-8-1 [3.2880]CLOSE-A-8-1 [3.6920]ARRIVE-A-7-1 [3.6950]OPEN-A-7-1 [3.9110]OUT-1-A-7-1 [4.1280]CLOSE-A-7-1 [4.6260]ARRIVE-B-2-7 [4.6280]OPEN-B-2-7 [4.8440]IN-2-B-2-7 [5.0470]CLOSE-B-2-7 [5.2520]ARRIVE-C-2-7 [5.2540]OPEN-C-2-7 [5.4690]OUT-2-C-2-7 [5.6850]CLOSE-C-2-7 [9.4520]ARRIVE-E-6-8 [9.6540]ARRIVE-B-1-9 [9.6570]OPEN-B-1-9 [9.6560]ARRIVE-D-6-8 [9.6570]OPEN-D-6-8 [9.8720]IN-4-B-1-9 [9.8740]IN-3-D-6-8 [10.0900]CLOSE-B-1-9 [10.0760]CLOSE-D-6-8 [10.2940]ARRIVE-C-1-9 [10.2810]ARRIVE-E-6-8 [10.4970]ARRIVE-D-1-9 [10.4990]OPEN-D-1-9 [10.4990]ARRIVE-A-6-8 [10.5000]OPEN-A-6-8 [10.7020]OUT-3-A-6-8 [10.7140]OUT-4-D-1-9 [10.9180]CLOSE-D-1-9 [10.9190]CLOSE-A-6-8	每个电梯分别处理对应楼层（楼座）的请求

说明：

- 1、为了方便说明，指导书上提供的输入为这样的格式：`[x.x]yyyyyyyy`。
- 2、意思是在 `x.x` 这个时刻（相对于程序运行开始的时间，单位秒），输入 `yyyyyyyy` 这样的一行数据。
- 3、关于上面说到的时间计测同步性问题，在这里解释一下：

- (1)、直观地说，可能会观察到输入指令时间晚于做出反应的时间或反应时间明显晚于预期时间的现象。
- (2)、这个实际上不是程序的错误，而是由于评测机投放数据的系统、和程序输出接口，这两边的时间可能存在不同步导致的。
- (3)、**这一问题是由于多线程测试不可避免的波动性导致的计时同步性误差，属于正常现象，不会被认定为错误。**（~~当然，也不会出现故意提早的情况，因为程序本身就是实时交互，不可能做得到预知未来~~）但是，电梯的操作还是应当遵从逻辑，不能出现乘客先进电梯，再开门这样的操作，**你的程序所输出的操作的时间戳也应当是递增的。**

第五部分：测试数据限制说明

一、数据基本限制

- 1、可输入电梯系统的指令数：**1~70条**（指令数为0或超过70均为不合法输入）。
- 2、系统时间上限 T_{max}：电梯系统时间 T_{real} 和电梯系统运行花费总时间 T_{final} 的上限时间，**一旦超过将直接判定为TIME_LIMIT_EXCEED。**
- 3、数据保证**请求一定能够完成运输**，即不会在某个楼层没有对应电梯时给出该楼层的横向运输请求。
- 4、对于初次增加电梯的楼层，确保在 1s 后才会出现对该楼层的请求。
- 5、电梯 id 和乘客 i 满足为 int 类型且 $id > 0$

二、公测数据限制

- 1、运行基准时间 T_{ALS}：基于 ALS 的纵向横向电梯调度策略运行的时间。
- 2、系统时间上限 T_{max}：计算方式为 $T_{max} = \max\left(T_{ALS} + 3, 1.05 \cdot T_{ALS}\right)$
- 3、包括强测在内的数据都会保证正常程序的电梯系统时间不会超过 T_{max}，也会**避免使用对边界情况较为敏感的数据。**
- 4、提示：**对于中测和强测数据，T_{max} 将会严格限制为通过上述公式计算得到的值。**~~所以，请不要试图用超长的 sleep 来回避线程安全停止的问题。~~

三、互测数据限制

- 1、系统时间上限 T_{max}：对于互测数据，T_{max}为210s。
- 2、第一条指令的投喂时间在 1s 或 1s 以后。
- 3、最后一条指令的输入时间不晚于70s。
- 4、指令条数不超过70。
- 5、要求每座、每层最多3个电梯，总电梯数量不超过 15。
- 6、**随测试点提交的输出**中电梯最终的关门时间 T_{final} 不得大于150s。

第六部分：公测说明

一、正确性判断

- 程序被认定为正确当且仅当以下 3 个条件同时满足：
  - 1、**代码实现中不存在轮询。**轮询是一种非常占用 CPU 资源的行为，为了避免出现轮询的情况，**请使用 Wait-Notify 的方式编程。**同时，为了检查是否出现轮询的情况，总 CPU 时间限制为 10s，不满足该限制的程序会被判定为错误。
  - 2、**电梯的性能符合基本要求**，即你的程序需要满足 $T_{real} \leq T_{max}$ $\&\&$ $T_{final} \leq T_{max}$。这里再次说明 T_{max} 的标准：
    - (1)、对于公测数据，T_{max} 取决于 T_{ALS}，计算方式为 $T_{max} = \max\left(T_{ALS} + 3, 1.05 \cdot T_{ALS}\right)$，基准为 ALS 调度策略。

- (2)、对于互测数据，T_{max} 为210s。

- 3、**输出结果符合逻辑**，具体来说，有以下几点：
  - (1)、电梯运行逻辑合理：
    - a、电梯到达一层后就可以输出开门信息，然后花 0.2s 开门，开门结束。
    - b、电梯开门结束后，某一刻电梯花 0.2s 关门，然后输出关门信息，就可以离开楼层。
    - c、电梯在两楼层间若发生移动，应满足移动时间要求
    - d、电梯只在可到达楼层运行。
  - (2)、人员进出逻辑合理：
    - a、只有已经在电梯里的人可以出电梯，也只有不在电梯里的人可以进入电梯。
    - b、如果电梯在楼座 A 楼层 2，那么乘客显然也只能在楼座 A 楼层 2 进出。
    - c、**反常识设定**：开门的 0.2s 期间或关门的 0.2s 期间乘客仍然可以进出，即在**整个开关门窗口时间内**乘客都可以进出。~~纸片人没有厚度~~
    - d、乘客进出信息在电梯开关门窗口时间之内。

      ```
      错误示例：
      [    4.6280]OUT-1-A-1 ← 这个人没开门就出来了
      [    4.8440]OPEN-A-1
      [    5.0470]CLOSE-A-1
      ```

    - e、 电梯系统结束运行时，所有的乘客都到达了目标楼层，且没有被困在电梯里。
  - (3)、电梯载客量合理：
    - a、**电梯任何时候，内部的人数都必须小于等于轿厢容量限制。**
    - b、也就是说，即便在 OPEN、CLOSE 中间，也不允许出现超过容量限制的中间情况。
  - (4)、时间戳合理：即**输出的时间戳应该是非减的。**

二、性能分的评判

在强测时，将电梯系统运行花费总时间 T_{final} 作为运行时间 time，计算相应的性能分。
设全体通过该数据点的同学的运行时间的平均值为 mean，标准差为 std。
设 $x = \frac{\mathrm{mean}-\mathrm{time}}{\mathrm{std}}$，则通过多项式拟合，性能百分比为

$$r(x) = 100\% \cdot \begin{cases} 1 & x \geq 1 \\ 0.17361111x^3 - 0.05952381x^2 + 0.73710317x + 0.15119048 & -0.2 < x < 1 \\ 0 & x \leq -0.2 \end{cases}$$

三、分数判定和组成

本次作业功能分为 85 分，性能分为 15 分，二者之和为总分。

第七部分：互测说明

- 1、**程序在互测中的正确性判断规则与公测说明中的一致。**
- 2、所有提交的互测数据均需要满足以下输入格式（该格式与样例一致，可参考样例中的输入格式）：
  - (1)、需要额外附加时间戳 `[time]`，其中time是一个保留一位小数的浮点数，比如 `0.0`,`1.5`,`2.2` 等。
  - (2)、每行的输入为乘客请求，其格式为 `[<time>]<id>-FROM-<A>-<x>-TO-<B>-<y>`。
  - (3)、评测系统将使用请求发射器在对应的时间将请求送入标准输入。

- 3、所有提交的互测数据均需要满足数据基本限制和互测数据限制。

第八部分：提示和警示

一、提示

- 1、关于接口的一些使用，可以在 idea 里面，按 Ctrl+Q 。将可以看到类和方法的使用格式以及注释（本次官方提供的接口均提供了中文版 javadoc 注释，Ctrl+Q 可以直接查看）
- 2、请**在代码一开始就初始化时间戳**，否则可能会出现和评测不一致的情况。

```
import com.oocourse.TimableOutput;

public class MainClass {
  public static void main(String[] args) {
    TimableOutput.initStartTimestamp();  // 初始化时间戳
        /*
            代码部分
        */
  }
}
```

- 3、官方提供的输出包是**线程不安全**的，需要同学们**自行保证其被多个线程调用时输出符合逻辑。**
- 4、由于**多线程调度存在随机性**，在使用官方提供的投喂脚本或进行互测环节时，可能会出现**输出与官方评测机不一致、bug 无法复现的情况**，最终结果均**以官方评测机为准。**
- 5、如何进行线程安全设计：
  - 明确共享对象的状态和约束
  - 更高效率的进行并发访问（如使用读写锁）
- 6、线程协同架构选择：
  - 对请求的处理分段：分配给横/纵向电梯 → 分配给哪部电梯
  - 对共享队列的选择：整体队列、楼座（层）队列、电梯队列
- 7、工厂模式的使用：采用工厂模式来创建不同运行模式、运行策略的电梯。

二、警示

不要试图Hack评测机和官方接口，不要企图伪造时间戳，不要抄袭。如在互测中发现其他人的代码疑似存在上述行为，可向课程组举报。课程组感谢同学们为OO课程建设所作出的贡献。