第五次作业指导书

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**，熟悉线程的创建、运行等基本操作，熟悉多线程程序的设计方法

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，电梯可以在楼座内 $1-11$ 层之间运行。

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：上下行，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。

二. 电梯参数说明

- 1、可到达楼层：1 - 11 层
- 2、初始位置：1 层
- 3、数量：6 部
- 4、编号：6 部电梯，ID分别为 1 —6
- 5、移动一层花费的时间：0.4s
- 6、开门花费的时间：0.2s
- 7、关门花费的时间：0.2s
- 8、限乘人数：6 人

三. 电梯请求说明

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**出发楼层和目的楼层**，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ALS**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

对于每一个电梯，都采用 ALS 策略，即新增**主请求**和**被捎带请求**两个概念

- 1、主请求选择规则：
  - (1)、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
  - (2)、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
- 2、被捎带请求选择规则：
  - (1)、电梯的主请求存在

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/输出说明

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

每一行是一个乘客的到达信息。格式为： `[时间戳]乘客ID-FROM-起点层-TO-终点层`

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

四.样例

输入	输出
[9.6]1-FROM-4-TO-5	[10.6380]ARRIVE-2-1 [11.0480]ARRIVE-3-1 [11.4540]ARRIVE-4-1

	[11.6610]IN-1-4-1 [11.8630]CLOSE-4-1 [12.2720]ARRIVE-5-1 [12.2730]OPEN-5-1 [12.2740]OUT-1-5-1 [12.6770]CLOSE-5-1
[4.1]1-FROM-2-TO-9 [7.5]2-FROM-1-TO-10	[5.0700]ARRIVE-2-1 [5.0710]OPEN-2-1 [5.2880]IN-1-2-1 [5.4860]CLOSE-2-1 [5.9010]ARRIVE-3-1 [6.3060]ARRIVE-4-1 [6.7190]ARRIVE-5-1 [7.1540]ARRIVE-6-1 [7.5670]ARRIVE-7-1 [7.9810]ARRIVE-8-1 [8.0500]OPEN-1-2 [8.2580]IN-2-1-2 [8.3950]ARRIVE-9-1 [8.4000]OPEN-9-1 [8.4010]OUT-1-9-1 [8.4650]CLOSE-1-2 [8.8080]CLOSE-9-1 [8.8860]ARRIVE-2-2 [9.3020]ARRIVE-3-2 [9.7120]ARRIVE-4-2 [10.1290]ARRIVE-5-2 [10.5370]ARRIVE-6-2 [10.9490]ARRIVE-7-2 [11.3530]ARRIVE-8-2 [11.7610]ARRIVE-9-2 [12.1690]ARRIVE-10-2 [12.1690]OPEN-10-2 [12.1720]OUT-2-10-2 [12.5950]CLOSE-10-2
[1.9]2-FROM-10-TO-6 [6.6]1-FROM-3-TO-5	[2.9360]ARRIVE-2-1 [3.3490]ARRIVE-3-1 [3.7610]ARRIVE-4-1 [4.1700]ARRIVE-5-1

	[4.9810]ARRIVE-7-1 [5.3970]ARRIVE-8-1 [5.8100]ARRIVE-9-1 [6.2230]ARRIVE-10-1 [6.2260]OPEN-10-1 [6.4330]IN-2-10-1 [6.6320]CLOSE-10-1 [7.0480]ARRIVE-9-1 [7.4580]ARRIVE-8-1 [7.6460]ARRIVE-2-2 [7.8670]ARRIVE-7-1 [8.0530]ARRIVE-3-2 [8.0530]OPEN-3-2 [8.2570]IN-1-3-2 [8.2770]ARRIVE-6-1 [8.2770]OPEN-6-1 [8.2780]OUT-2-6-1 [8.4570]CLOSE-3-2 [8.6920]CLOSE-6-1 [8.8750]ARRIVE-4-2 [9.2850]ARRIVE-5-2 [9.2850]OPEN-5-2 [9.2870]OUT-1-5-2 [9.6990]CLOSE-5-2
[2.7]1-FROM-2-TO-8 [4.3]2-FROM-11-TO-3 [7.4]3-FROM-3-TO-8	[3.7670]ARRIVE-2-1 [3.7680]OPEN-2-1 [3.9750]IN-1-2-1 [4.1780]CLOSE-2-1 [4.5910]ARRIVE-3-1 [5.0060]ARRIVE-4-1 [5.3560]ARRIVE-2-2 [5.4160]ARRIVE-5-1 [5.7640]ARRIVE-3-2 [5.8240]ARRIVE-6-1 [6.1730]ARRIVE-4-2 [6.2330]ARRIVE-7-1 [6.5790]ARRIVE-5-2 [6.6420]ARRIVE-8-1 [6.6430]OPEN-8-1

	[7.0520]CLOSE-8-1 [7.4140]ARRIVE-7-2 [7.8390]ARRIVE-8-2 [8.2810]ARRIVE-9-2 [8.4690]ARRIVE-2-3 [8.7000]ARRIVE-10-2 [8.8790]ARRIVE-3-3 [8.8820]OPEN-3-3 [9.0920]IN-3-3-3 [9.1190]ARRIVE-11-2 [9.1220]OPEN-11-2 [9.2920]CLOSE-3-3 [9.3390]IN-2-11-2 [9.5280]CLOSE-11-2 [9.7090]ARRIVE-4-3 [9.9480]ARRIVE-10-2 [10.1180]ARRIVE-5-3 [10.3530]ARRIVE-9-2 [10.5250]ARRIVE-6-3 [10.7600]ARRIVE-8-2 [10.9270]ARRIVE-7-3 [11.1780]ARRIVE-7-2 [11.3320]ARRIVE-8-3 [11.3340]OPEN-8-3 [11.3410]OUT-3-8-3 [11.5850]ARRIVE-6-2 [11.7410]CLOSE-8-3 [11.9970]ARRIVE-5-2 [12.3980]ARRIVE-4-2 [12.8090]ARRIVE-3-2 [12.8100]OPEN-3-2 [12.8100]OUT-2-3-2 [13.2240]CLOSE-3-2

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：

- (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：限制说明

一.数据基本限制

- 可输入电梯系统的指令数：**1~100条**（指令数为0或超过100均为不合法输入）。
- 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。T_{max}与标程运行时间T_{std}的计算公式在下文给出。
- 乘客ID保证不重复，存在于电梯系统内的电梯ID保证不重复，所有ID均为int范围内正整数。
- 乘客请求中保证起点层和终点层不同
- 输入数据的时间戳精确到**小数点后一位。**

二.公测数据限制

- 标程运行时间T_{std}：课程组标程运行测试点所使用的时间。
- 公测中T_{max}与标程运行时间T_{std}的计算公式如下。

$$T_{max} = \max\left(T_{std} + 10, 1.15 \times T_{std}\right)$$

- 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。**
- **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

- 系统时间上限 T_{max}：对于互测数据，T_{max} 为120s。
- 第一条指令的投喂时间在1s或1s以后。
- 最后一条指令的输入时间不晚于50s

第六部分：公测说明

一. 正确性判断

- 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 $\max(T_{real}, T_{final}) \leq T_{max}$。
- 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、电梯只在大楼的楼层范围内运行。
  - e、电梯系统结束运行时所有电梯必须关着门
  - (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、如果电梯在楼层 2，那么乘客显然也只能在楼层 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-1-1 ← 这个人没开门就出来了
[   4.8440]OPEN-1
[   5.0470]CLOSE-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

- 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，性能分将由三部分组成：系统运行时间，等待时间与期望时间之差的最大值和系统耗电量。其中，系统运行时间即为$T_{run} = \max T_{real}, T_{final}$。最大等待时间与期望时间之差计算方法如下。

对第i个请求，设其等待时间为$T_i = T_{\text{到达目标楼层}} - T_{\text{发出请求}}$，该请求的期望等待时间$ET_i$计算公式如下。

$$ET_i = \frac{\sum_{f=F_{min}}^{F_{max}} |F_s - f| T_{move}}{F_{max} - F_{min} + 1} + T_{open} + T_{close} + |F_d - F_s| T_{move}$$

$$F_{min}：电梯系统最低楼层$$
$$F_{max}：电梯系统最高楼层$$
$$F_s：请求的起始楼层$$
$$F_d：请求的目标楼层$$
$$T_{open}：开门时间$$
$$T_{close}：关门时间$$
$$T_{move}：电梯移动时间$$

本次作业，上述公式中为定值的参数如下。

$$F_{min} = 1$$
$$F_{max} = 11$$
$$T_{open} = 0.2$$
$$T_{close} = 0.2$$
$$T_{move} = 0.4$$

整个公式中，后三个数是电梯将该乘客送往目的楼层的所需时间，第一个项是在假设电梯在各楼层的概率分布为均匀分布时电梯到达起始楼层的期望。则等待时间与期望时间之差的最大值为

$$MT = \max_i (T_i - ET_i)$$

系统耗电量将使用以下方法计算。

根据输出信息中的 `ARRIVE`，`OPEN`，`CLOSE` 信息的数量进行计算。

- 系统的耗电量$W = W_{OPEN}N_{OPEN} + W_{CLOSE}N_{CLOSE} + W_{ARRIVE}N_{ARRIVE}$，其中$N_{OPEN}, N_{CLOSE}, N_{ARRIVE}$分别表示输出中$OPEN, CLOSE, ARRIVE$信息的总数。

设对于所有同学的数据，参数x的平均值为x_{avg}，最大值为x_{max}，最小值为x_{min}，则定义以下折算函数

$$base_{min} = p \cdot x_{avg} + (1-p) \cdot x_{min}$$
$$base_{max} = p \cdot x_{avg} + (1-p) \cdot x_{max}$$
$$p = 0.25$$
$$r(x) = 100\% \cdot \begin{cases} 1 & x \le base_{min} \\ 1 - 10^{1-\frac{base_{max}-base_{min}}{x-base_{min}}} & base_{min} < x \le base_{max} \\ 0 & x > base_{max} \end{cases}$$

最终性能分s将由以下公式计算。

$$s = 15 \times (0.3r(T_{run}) + 0.3r(MT) + 0.4r(W))$$

其中T_{run}, MT, W为你的程序的运行时间，等待时间与期望时间之差的最大值和系统耗电量。

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：提示与警示

提示

一、实现提示

- 1、电梯的"运行策略"实际上可以抽象为从一组待选任务当中选取一个来执行。可以先编写一个**候乘表类**来显示当前电梯系统中各层已有的乘客。再编写一个**策略类**，接受这个候乘表类，分析并输出一个目标位置。因此，分析的过程被隐藏在了策略类内部。
- 2、将策略类作为电梯的一个属性，通过更换策略类，就可以灵活地切换执行策略。或应用工厂模式组装不同电梯。
- 3、电梯的运行流程可以抽象为：到达一层→开门→乘客进出→关门→寻找下一目标层→移动，可以使用状态模式建模。

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.elevator1.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**线程安全**的。
- 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、性能分公式可能比较复杂，但实际上公式表达出的意思就是希望你的程序**在保证正确性的前提下**能够尽量做到以下三点。
  - 运行时间尽量短
  - 尽量不让请求等待过长时间
  - 尽量减少系统的无效运行

警示

不要试图Hack评测机，不要抄袭。如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课程建设所作出的贡献。