

第三次作业的 BUG:

第三次作业中由于笔误，在电梯向下运行分支的输出语句处少写了个“-1”，导致有几个测试点没有通过。

```
Output().output(e_waitlist.getReq(i), elevator.getCurfloor(), ElevatorState.DOWN, elevator.getE_time(-1));
```

程序功能说明:

程序使用多线程实时模拟三部电梯的运行。程序从控制台中即时读取若干条对电梯的请求，以“END”表示程序模拟结束结束，输出即时的电梯相应请求时地运动状态信息到文件 result.txt 中。

程序运行所需环境和运行指令规范:

本程序需要在 java 虚拟机环境下运行，电梯请求从控制台输入，输出结果保存在工程目录下的 result.txt 文件中。

运行指令为:

```
javac ElevatorSys.class
```

```
java ElevatorSys
```

程序的输入说明:

除指导书中对输入的要求外，本程序对其未明显规定的输入规定如下:

- 1、本程序不支持楼层号、电梯编号前出现前导 0 或者“+”，如若出现，该请求被当做 INVALID 请求处理。
- 2、程序规定退出程序的命令为“END”，为大写。一旦输入命令“END”并按下回车后，程序会立即强制结束，可能会导致部分请求的相应信息还没有输入到 result.txt 中。
- 3、对于在电梯进程执行代码的过程中加入的新请求，由于进程切换以及代码运行的所耗时间的原因，可能会在边界测试时发生不可预测的结果。

程序计算结果的输出规格:

- 1、输出中的 st 是指系统时间，从 1970 年到指令输出所经过的毫秒数。t 或者 T 则是以秒为单位，保留一位小数，表示从输入第一条指令开始(无论有效或无效)，到指令输出或者输入时的秒数。
- 2、本程序逻辑判断中涉及的时间都没有取整，只有在最后输出的时候相关时间才取整，并保留一位小数。
- 3、由于程序代码及线程切换的误差，可能导致程序对电梯请求输出的时间间隔不完全是 3.0 或 6.0，可能会多 0.1 或 0.2 秒，如果出现这种情况，请多测几次，实在不行那也没办法了。

本程序的一些特性:

- 1、对于同一行的请求，虽然他们的请求出现时间是一样的，但是本程序仍然按照从左到右的顺序依次分配这些请求，并且每个请求被分配后立即改变电梯的状态，所以分配结果可能与测试者预期不符。
- 2、如果一行中有多个请求无法立即响应，本程序则将不能响应的请求放入专门的不能响应请求等待队列中，这种操作可能会导致本来先响应的请求在后面才能响应(也就是虽然请求到达时间一样，但可能右边的请求优先于左边的请求被响应)。

3、本程序的主请求在到达主请求目标楼层开关门时刻之后才更换，所以开关门期间在判断一个新请求能否被该电梯捎带时使用的是旧主请求。

比如：

```
0s=(ER,#3,20)
20s=(ER,#1,3);(ER,#2,2)
22s=(ER,#1,5)
29s=(FR,4,UP)
34s=END
```

这样的测试，(FR,4,UP)发生时，#1 电梯正在开关门，旧的主请求不能捎带该请求，而新的主请求可以捎带此请求，按照本程序的特性，#1 电梯将不响应(FR,4,UP)请求，所以(FR,4,UP)请求被处于 WFS 状态的#2 所响应。

4、本程序在电梯临时停靠开关门时（不是同层请求那种开关门，电梯开门之前是运动的）是可以捎带符合条件的请求的。

5、本程序在分配一个之前无法响应的请求时，若此时刻同时(在毫秒级上是一样的)有多个电梯完成之前的任务，可以捎带这个新的请求，则理论上该请求应该分给运动量最小的，但是本程序由于线程切换的先后顺序等不可控问题，可能分配给一个运动量不是最小的电梯。如示例：18.0 时三个电梯都能捎带(FR,10,UP)请求，但是调度器却分给了二号电梯，而没有分给 1 号电梯。

```
(ER,#1,1);(ER,#1,2);(ER,#1,3);(ER,#2,7);(ER,#3,7);(FR,10,UP)
```

```
1526923729749:[(ER,#1,1), 0.0] / (#1, 1, STILL, 0, 6.0)
1526923732748:[(ER,#1,2), 0.0] / (#1, 2, UP, 1, 9.0)
1526923741757:[(ER,#3,7), 0.0] / (#3, 7, UP, 6, 18.0)
1526923741757:[(ER,#2,7), 0.0] / (#2, 7, UP, 6, 18.0)
1526923741757:[(ER,#1,3), 0.0] / (#1, 3, UP, 2, 18.0)
1526923756749:[(FR,10,UP), 0.0] / (#2, 10, UP, 9, 33.0)
```

为了解决这个问题，本程序采取了一个小 trick，在大部分时候（自己测试还没发现错误）可以解决上面的问题，如果没有非配给运动量最小的，请测试者重新测试。

运行错误响应信息：

程序运行过程中可能会在控制台输出一些莫名的东西，比如：















```
"regex error!"
```










```
Input error!
```








```
Pipe error!
```

等，这说明我的程序出现了 bug，并且我自己没测出来。







程序类图：

- ▼  oo3
 - >  Autotest.java
 - >  Direction.java
 - >  Elevator.java
 - >  ElevatorState.java
 - >  ElevatorSys.java
 - >  Floor.java
 - >  NewScheduler.java
 - >  Output.java
 - >  ReqType.java
 - >  Request.java
 - >  RequestList.java
 - >  Scheduler.java
 - >  UncaughtException.java

- ▼  ElevatorSys
 -  FLOOR_MIN : int
 -  FLOOR_MAX : int
 -  printer : PrintWriter
 -  startTime : long
 -  main(String[]) : void
 -  isValidReq(String) : boolean
 -  string2Req(String, long) : Request
 -  getStartTime() : long
 -  isDebug() : boolean

- ▼  NewScheduler
 -  reqList_all : RequestList
 -  floor : Floor
 -  printer : PrintWriter
 -  ele1 : Elevator
 -  ele2 : Elevator
 -  ele3 : Elevator
 - >  selectQueue : PriorityBlockingQueue<
 -  NewScheduler(RequestList, Elevator, E
 -  dispatchOneFR(Request) : boolean
 -  run() : void

- ▼  Elevator
 - ele_id : int
 - floor_class : Floor
 - printer : PrintWriter
 - state : ElevatorState
 - curfloor : int
 - e_dir : Direction
 - e_time : long
 - targetfloor : int
 - e_button : Boolean[]
 - aom : int
 - ▲ df : DecimalFormat
 - FRshutdownList : ArrayList<Request>
 - ERshutdownList : ArrayList<Request>
 - waitQueue : BlockingQueue<Request>
 - > ▫ upQueue : PriorityBlockingQueue<Request>
 - > ▫ downQueue : PriorityBlockingQueue<Request>
 - curQueue : PriorityBlockingQueue<Request>
 - tricktime : long
 - ^c Elevator(int, Floor, PrintWriter)
 - goUp(int) : void
 - goDown(int) : void
 - gowait(double) : void
 - goOpenClose() : void
 - getEle_id() : int
 - setEle_id(int) : void
 - getE_dir() : Direction
 - setE_dir(Direction) : void
 - getTargetfloor() : int
 - setTargetfloor(int) : void
 - setState(ElevatorState) : void
 - setCurfloor(int) : void
 - setE_time(long) : void
 - getCurfloor() : int
 - getE_time() : long
 - getAom() : int
 - setAom(int) : void
 - pushButton(int) : Boolean
 - popButton(int) : Boolean
 - isButtonOn(int) : Boolean
 - ^o receiveReq(Request) : void
 - ^o canResponse(Request) : boolean
 - ^o canShaodai(Request) : boolean
 - [▲] run() : void

- ▼  RequestList
 - ReqList : BlockingQueue<Request>
 - nextindex : int
 - ▲  RequestList()
 - pollReq() : Request
 - getReq(int) : Request
 - addReq(Request) : void
 - deleteReq(int) : void
 - deleteReq(Request) : void
 - getListlen() : int
 - getNextindex() : int
 - setNextindex(int) : void
 - nextindexadd1() : void
- ▼  Request
 - type : ReqType
 - floorNo : int
 - FRdir : Direction
 - ele_id : int
 - time : long
 - status : int
 -  Request(ReqType, int, Direction, long)
 -  Request(ReqType, int, int, long)
 - getEle_id() : int
 - setEle_id(int) : void
 - getStatus() : int
 - setStatus(int) : void
 - getType() : ReqType
 - getFloorNo() : int
 - getFRdir() : Direction
 - getTime() : long
 - setType(ReqType) : void
 - setFloorNo(int) : void
 - setFRdir(Direction) : void
 - setTime(long) : void
 -  toString() : String
 - toString3() : String
 - toString2() : String