出租车说明文档

终于要结束了 xixi

基本要求符合指导书规范。

这里直接根据设计要求来阐述整个程序的要求：

1、叫车请求队列采用 ArrayList 实现，所以应当不会有队列容量的问题。

2、程序通过控制台获取乘客请求，请求格式为[CR,src,dst]，其中 src 和 dst 均为(i,j)形式的坐标位置，请求可以带有过滤空格（不包括制表符），i,j 为两位以内的整数（包含前导 0），不支持正负号，格式错误以及超出地图范围的坐标会被直接忽略，出发点和目的地相同的请求也会被忽略，输出”Wrong Format”或者”Out of range”，请求产生时间取自系统时间对 100ms 取整，若两请求的 src 和 dst 相同，且产生时间对 100ms 取整后相同，则判为同质，输出”SAME”并忽略后输入的请求。不允许一行多条请求。

3、对请求的处理过程会输出到三个文件中：这里原本是 D 盘下的 ReqInfo.txt，WinInfo.txt，MoveInfo.txt，如果你没有 D 盘，请更改 Main.java 中的文件路径。

各文件包括的内容如下：

ReqInfo：所有的有效请求内容：发出时刻、请求坐标、目的地坐标

格式：Time:发出时刻 SRC 请求坐标 DST 目的地坐标

例：Time:1524593636800 SRC(20,24) DST(38,41)

WinInfo：接单窗口关闭时，对应请求的所有参与抢单的出租车信息：车辆编号、位置、状态、信用

格式：请求 Window at 派单时间（窗口关闭时间）

Taxi:车辆编号 Locatio:位置 Credit:信用 Status:状态

...

Dispatch to:Taxi 抢到该单的出租车编号

例：

[CR,(20,24),(38,41)] Window at 1524593639800

Taxi:34 Location:(23,19) Credit:1 Status:WAITING

Taxi:18 Location:(19,23) Credit:1 Status:WAITING

Dispatch to:Taxi 18

MoveInfo：被派单的车辆的运行信息，以 500ms 为间隔，从接单到到达目的地为止，输出其经过的分支点坐标和经过时刻。注意：首先指导书中要求的派单时车辆坐标，派单时刻，乘客位置坐标，目的地坐标均在 WinInfo 中有记录。其次达到乘客位置时刻为其连续 2 个时间片（1s）未改变坐标的首条记录时间，即会出现 3 条输出的坐标相同，这 3 条的第一条时间就是到达时间，到达目的地时刻在输出时会特别的输出“DST---”前缀。请自行查找。

格式：Taxi:车辆编号 arrive at 分支点坐标 at 经过时刻

（到达乘客位置和到达目的地均为这个记录格式）

例 Taxi:18 arrive at (20,23) at 1524593640000

到达时刻为：

DST---Taxi:18 arrive at (20,23) at 1524593640000

4、出租车查询状态信息接口

位于 TaxiSquad.java 中的 GetInfo()方法

输入参数为车辆 id，范围 0-99，可以于控制台输出查询时刻，出租车当前坐标和当前所处状态。

请于测试线程内进行调用查询。

5、按状态查询出租车接口

位于 TaxiSquad.java 中的 GetTaxiOf()方法

输入参数为状态，可选{TaxiStatus.STOP, TaxiStatus.TAKING, TaxiStatus.SERVING, TaxiStatus.WAITING}中的一个，会在控制台输出查询时刻所有处于相应状态的出租车编号。

请于测试线程内进行调用查询。

6、GUI 包的内容进行过修改，并且本程序中调用了一些方法，不允许测试者修改 gui.java 中的内容。设计者已经去掉了红框和最短路径弹窗。

特殊说明：

1、 程序会持续运行直到关闭 GUI 界面，建议待出租车跑单完毕再关闭程序，否则输出可能不完整。

2、 地图信息默认从 D:\Map.txt 读取（在不使用 loadifle 的情况下），请确保此文件的存在和有效。若 Map.txt 存在问题，例如每行的数字个数不为 80 个以及非法数字等等，均会输出"Invalid input in Map.txt"，若格式正确，但是图不连通，则会输出："地图并不是连通的,程序退出"，注，地图信息的正确性请测试者依照指导书保证一切不依照指导书输入的地图信息均有可能使程序报错并自行结束。

3、 使用 loadfile 的流程如下：首先运行程序会在控制台输出" Please input "LOAD" for loadfile, other input to cancel loadfile"，如果使用 loadfile 需要输入大写 LOAD，然后输入绝对路径进行读取，并且对地图，出租车位置，状态，信用，道路流量，初始请求进行初始化，中途发生异常如没有输入 LOAD，或地址不存在，会读取默认地图。地图格式为一个 80*80 的矩阵。信号灯的初始化

和指导书相符，并且倘若在不能放置红绿灯的位置放了灯，会给出提示并且忽略该信号灯。出租车初始化为使用形如“id,status,credit,(x,y)”的方式进行初始化，例如“2,TAKING,0,(3,3)”这里需要解释在对出租车指定为接单状态 TAKING 以及服务状态 SERVING 的情况下，由于此时不可能有接单或服务的情况，出租车会自动转化为 WAITING 状态，这个已经咨询过助教是可以的，信用初始化请保证在 100000 以内。流量初始化按照形如“(x1,y1),(x2,y2),flow”的方式进行初始化，例如“(0,0),(0,1),2”，请保证 flow 在 int 范围内，坐标在合理范围内。请求按照标准形式进行输入，此处不允许空格，其他要求同上面的请求格式。需要注意的是，此文件的一切格式都需要测试者保证，由于各种格式错误，越界，小数，重复请求等错误所带来的程序退出，输出有误均不应当被申报为 bug。程序包中附了一个 a.txt 的文件可以供测试者参考

4、 若地图信息没有问题，则会进行初始化，GUI 界面出现后输出 Ready to Input，之后才可以在控制台输入请求，在提示信息输出前的所有输入均不会被记录。

5、 空闲出租车的随机移动策略为从可选的路径中（上下左右）随机选择流量最小的移动。

6、 派单时的随机策略是（在信用和距离相同的情况下）分配该单给最早接单的出租车。

7、 派单时倘若某辆出租车由于触发等待 20s 停止 1s，则不会派单给这辆出租车，即派单严格遵循只派单给等待状态的出租车。

8、 接单窗口结束时刻才进入范围的出租车抢单操作无效。

9、 出租车策略是每走一步进行一次 bfs 搜寻最短路径中流量最小的走。

10、 有关开关道路，通过从控制台输入相关命令进行控制，输入格式为[OPEN,X,Y]以及[CLOSE,X,Y]，允许空格，其中 X,Y 为 0 到 6399 之间的整数，是地图中某

个点的序号，加前导零共不能超过 4 位，例如点（A,B）对应的序号为 A*80+B，其中 A,B 均为 0 到 79 之间的整数。不能开启已开启的道路，不能关闭已关闭的道路，不能开启本来没有开启的道路，即可以对一条本来开启的道路关闭，再开启。由于在车辆行驶过程中开启，关闭道路会导致奇怪的事情发生，比如影响 bfs 过程，程序是在每次有车辆行动后更新道路情况，即把输入的请求放入一个数组，在间隙执行所以可能出现[OPEN,0,1],[CLOSE,0,1]中第二条失败的情况，是由于第二条输入时第一条还没有执行，属于正常情况。一切输入指令的正确性请测试者保证，并且遵循指导书要求。

11、 关于测试线程

有一个名为 TestThread 的类中提供了示范来调用这两个方法，测试者可以自行发挥。使用时请把 Main 里面的有关注释去掉。需要注意的是，由于系统时间是从第一条有效请求开始算的，所以在输入请求之前，函数调用输出的时间仅仅为模拟时间。

12、 关于 JSF

尽力了兄弟，手下留情

13、 关于 repOK

基本所有的类都写了 repOK 方法，但是看课件也没太看懂，只能粗略的写了一些，大概就是判断合法性，比如 taxi 类的 repOK 判断其所在位置，状态是否合法等等，其他的真的写不出来了，感觉写一下判断某个对象是不是某个对象也没有什么意义，希望兄弟手下留情！另外枚举类没写 repOK，感觉没啥用 ···

14、 关于红绿灯

由于指导书有歧义本程序的红绿灯在初始的时候是互相独立随机的，但是是一同变

化的，这点助教说了可以 readme。本程序对红绿灯处理如下。当遇到红灯时，记录所走方向，倘若下次应当移动时（此时已经变灯）这条路仍然存在，那么就走这条路，倘若道路被拆除了，边从能走的其他路中随机选一条行走，不受当前红绿灯约束。由于红绿灯的存在，输出文件 MoveInfo 中一辆车的变化时间会在 500-1500 之间变化，是正常的。

15、 LSP 论证

本程序的 SuperTaxi 继承自 Taxi 类，在基础上仅仅增加了一个 ArrayList 用来保存其执行过的请求，显然不会对父类对象的行为造成影响。子类新增了迭代器及相关方法，子类继承自父类，其父类方法是通过判断子类类型进行的相应处理，所以倘若进行替换时，例如改为 100 个可追踪出租车，其相应的 move 方法均会自动进行对应，选择，其需要的信息也会自动记录。子类由于完全继承了父类的方法和属性，新增的请求保存不一定有效，例如还没有请求，所以 repOK 是完全一样的。所以可以知道，由于方法的全部继承，所有出现 Taxi 的地方都可以用子类 SuperTaxi 替换，且不会让调用父类型的客户程序从行为上有任何改变。

16、 可追踪出租车相关

新增的可追踪出租车是粉色的且不会改变(由于 GUI 原因),可以通过关闭的道路。但是其实是有四个状态的,可以保存其被派到的所有订单信息,包括请求相关信息,出租车从抢到单到完成订单的所有经过路径点。通过每次服务进行管理。查询方法通过控制台实现，内部是迭代器。在控制台输入“Iterator”进入迭代器查询环节，然后输入查询的出租车 id，这里根据指导书要求是 0-29 之间的整数，然后输入 Next 可以通过迭代器依次向后查询历史，Last 向前查询历史，当迭代器到达边界会输出“Reach Bound”,Close 退出迭代器，然后便可以继续输入请求。这里

需要要求测试者保证其输入格式的正确。

再一次谢谢你阅读我的readme测试我的代码，大家写的都不容易，手下留情好人有好报！

祝你永不脱发