放在前面：

本程序将100个出租车放入一个线程进行运行。但是也满足了多线程要求（经过询问学长，这种方法是允许的，请同学不要误判，或误认为无效等，感谢！另外，readme中没有提到的一些细节，请以指导书和issue为准。

测试环境：

请使在搭载 windows10 系统的非 MAC 系列电脑上 JDK 版本 jdk-8u171-windows-x64 环境 中，使用 eclipse-java-oxygen-2-win32-x86\_64 运行该程序。如果以上要求不满足，运行时 有可能出现未知错误。注：仅支持控制台输入，不支持使用文件输入（load属于文件调用）和命令行输入。

输入：

1. The system initing! 之后可以在sonsole输入Load 指令。
2. 系统加载地图等需要一定时间，请在"The system ready!"字样出现后，在输入请求。
3. Load指令在程序最开始进行初始化的时候在eclipse控制台输入使用且仅能使用一次，。如果其中的map部分出错会直接结束整个程序，其他的出错会进忽略。如果load文件名出错或不存在，会进行反馈，并进行等待，直到正确的load指令输入，再进行初始化。
4. Load的map必须是内容而不是文件。
5. 再load之后仍然可以通过输入请求来增加订单。
6. 再load之后仍然可以通过输入指令来查看车的状态。
7. CheckTaxi指令以$开头，后面直接跟数字。不$开头的进入后面的判断。
8. CkeckTaxi指令输出的是出租车上一次移动后的状态，第一个时间是现实时间，第二个时间是上一个运动周期的出租车时间。
9. Load指令请务必按照正确的格式输入，如果里面的文件输入格式错误，后面发生的bug可能无法预料，但不会crash，也有可能直接触发系统shut down。
10. Flow的输入请严格遵守(x1,y1) (x2,y2) value的格式即value和第二个点之前都有且仅有空格，括号内无空格。具体可以参见文末的样例。
11. 请求的输入支持前导0，不支持前导+和负数，但是请不要输入>=80或<0的位置坐标。
12. 出发点目的地一致的请求算作无效请求。触发INVALID。
13. 除非您计算机性能特别强大，否则请不要尝试一次运行太多指令，毕竟理论上有可能计算不完。
14. 如果想要关闭程序只需要将GUI的窗口点击关闭即可，程序会直接自动关闭。
15. 由于本次测试不测格式，所以不是所有的报错都会有输出，有可能直接忽略而不显示，也有一定可能直接触发系统结束。
16. 程序的所有输入数值请不要大于100000，不要有前导+。
17. 总请求数大于300时，输入线程结束

输出：

1. 输出时，为每个有效的（包括有无被派单的请求）建立一个日志（也就是一个文件）里面记录与该请求有关的信息。具体的路径为，与src文件夹和map.txt同级。
2. 输出格式请同学测试一组查看，格式很直观，恕时间紧张不再多做解释。大概顺序依次为：请求本身，曾经抢过该单的所有出租车，派单的情况，前往乘客的路程，搭上乘客前往目的地的过程，最终到达的时间
3. 输出的参与抢单的车的状态会被认为是指最后派单时候的状态。
4. 出租车移动过程在节点上中输出的状态，除了接到乘客后休息一秒结束的节点以外。均值的是在该节点进行完状态转换之后的状态
5. 出租车移动过程在节点上中输出的时间是绝对时间，所以都没有划归到100的倍数，其主要价值在于差值。
6. 乘客上車后（1s之后），才输出接到乘客的日志。到达乘客地点的时候不会产生日志。
7. 同质请求（100ms内的起点终点相同者）会算作一个请求计入输入的请求数，分派其一个ID，但是不会算数console中会输出同质提示，也不会为其建立日志。
8. 如果load的地方特殊，可能sonsole有一些输出优先于The system ready!。这从事情请不要计较，毕竟console本来也不在检查范围。
9. 幽灵GhostRequest所有车统一输出到request0.txt。但是由于整个过程不完整里面会缺失一些东西，请同学不要在意。
10. 为了防止信息输出不全，在request输出中，边界时间节点（状态进行变化）的输出会稍有一些累赘。边界状态也不好把握，请同学见谅。

出租车行为解释：

1. 休息时间不会被打断。
2. 出租车不会中途掉头，所以出租车的所有活动时间都在距离初始时间200的整数倍点上
3. 判断抢单范围时是如果在半路上，视为在原路口才进行抢单。
4. 当出租车仍在某条路上（非路口）的时候，如果被派单，将继续完成此路段（即不会突然掉头）再进入服务状态，也因此派单时间（窗口关闭时间）和最终到达的时间的差很可能不是200的倍数。
5. 如果出现幽灵乘客的现象，根据状态执行[CR,(20,0)(10,0)]请求，此条请求的编号为request0，在request0.txt中输出所有车辆的情况。（由于中途派单，故输出内容信息不完整。）同时，此条请求不存在分派等过程，永远不会消失。
6. 运行过程从道路塌陷，出租车将继续行驶至写一个路口。

SetRoadStatus行为解释：

1. SetRoadStatues时，请在console输入命令，格式请严格遵守#(x1,y1) (x2,y2) status的格式，即status和第二个点之前都有且仅有空，括号内无空格，具体可以参见文末的样例。
2. 只有第一个字符为‘#’时才会进入SetRoadStatues命令的判定，否则系统按照请求进行识别。输入不合法的时候会报错，同时请务必保证开闭之后整张图仍是连通的，否则会出现奇怪的错误，比如直接程序结束。（显然这已经超出题目的要求）
3. 新开通的路径不会影响已经进入服务状态的出租车的路线，其最短路径不会更新。
4. 车在接上乘客和接到的单发生效果的时候，以及碰到现有最短路径上出现断路的时候，才会重新计算最短路径。
5. 对于已经进入服务状态的出租车，如果其最短路径上某条路断了，只有当出租车到达该断掉的路径时，才会重新计算新的最短路。

派单行为解释：

1. 3s窗口结束后，派单时，如果一辆车处于等待状态之外的状态，即使他是最合适的或者唯一抢单的，也不会被派单。
2. 当信用和距离一致时随机选择队伍中的第一个（约等于随机）

派单行为解释：

1. GUI不能对Load产生的初始流量进行显示（其固有缺陷），但是出租车会对初始化的流量进行响应，即第一个500s内能够观察到初始化的流量。
2. GUI中不写JSF

写在最后：

人生艰难，OO不易，求同学手下留情OTZ。文明六系，和谐OO。

Load样例

#map.txt##

#map

12232113333333213231132313113223331111312212221323133331312333133111321233333312

11331123213122213131112112213221322111312311112111233222111313131112331322232112

31132223212321221121221313121323223112312223213132122123323311123212213233131332

12121112213111211132211323331212123331122232313211213113323132122223112222333332

31213222222313333122112122332322222113322132323333122113231112211123232221321312

31233321312123123332333223213332112333131333233312312213213211112131233223113112

21231311132132123213312311131121111323333213112312121311223222223331331313323222

12311123222231223313331222223211112311313311222132331132311233131323332223123312

12213313132221223331113312323221122223312311331132123122223221232111212112133322

22233231311221331213223323321113112211231333111111332332132222133313333231222112

23131133211231333131323323212122213313112122311222322112313312322322213222121132

13331122331322123221131322331222133132121232321311313222313212131312111313121122

11331111221313112123313312231132131311213113213231212233113132222122311333113122

31312311211312232123331132323322113313333332332231313232122233122331312121133232

33233332323122311212113223211123132121222311133221333231123221122322122321113122

32212333132312113132222112321321222323221321211321131123232312111111323323112212

21122212233221123123333112132131313112223222131113231212212321333222113332221232

33132312223213213221321211112113212123111111233132331312321111212232131321211322

32213321221322133331221113232311333133222311133132213122232111132212131321133212

22312231322131312322333211322123113231331313111212111323131333121211333231122212

11331123113332211123123312312311211311211311221211223121323122131123111131233132

32212212333321212222113323313231212322112131121322232312331213221313122323313132

12122231231322321133223123132232213231312122122122333312132212212312321323123212

11231312333122231112213221112222221212323223211121123213322321221322112231223132

22132131322333322131333313233211323121331323213212331331223312331131331323331332

23132322131333221212211111131311212322232212322312121311111323131211232313312322

33321133131133331232213221312211321131211323313121123113121211133322323122333112

33212133221313131213133332222213312322311311233321222332123232121133231233322212

32132212312132331313223132321223312231223122213211133123113112331311323333332322

12113221212113121222133121333211112123132321113321232331211223212133122123123212

33213322212332331112332211132323323133223121113222212311121311333221121313213232

22323311121121312221322223313223133313121233323111122323222213133311123233323212

12221132331311322323233213133331131321223321323222121211123111122121213131221312

21112123313313232211111121233111332313212322231111113321311332331113211332131312

32332113332313231333212133212233232231312111313223333132233313111322212311113222

11213112221121233212213232322231121333222231111211131212131221121223311212233212

13212323312133123233231321331222321222223211223111131322323212332111311313231232

23331221312123111211321311131331232132231221333222132312131311131132231123211332

11233123231332112313212321322323211133231223232112121223332132131122211122232232

13131211223213333111133332322323322232231212222221321133131333112311312331211112

31231111131231333223222133323232331233111321312221131211311211131111113121232112

23213112223223333111213332222122331112213332232322131112212123231312321121311332

32221331123133332323212221333312323333333333323323331232312322221312123312212112

21313111131232123311122321332322212312211111221322212132312132212211122112112122

31213232133131311312323331333212112211333323213233323122121312222312212322212132

11123113121222131222221212313211222222232322113113232231112323123221213112211222

33123113123123231233322113223323332112311222121232333233233332111313112233132112

23231123333221212323222331233211223132332331132132233333333331222211333122212232

22211131333221331112222332231312133132222122132111112122233332121122322113131132

23331223133232132321212221331112132112321232221213223132331332323121213332133312

33223332212211132112321112322321211212333131212223222131232113332133212332221222

11222122332222312313211131111123233322113221332311232331332222312232313311112112

21111321122122121222121122311123213312331213233131313222311112332313221222131312

11212313121233222112231213313133113223222221232232311332311131221132121222121122

11212111333213311211211213123222212323123322112331133321132323332311311133233122

22121221132133233323321222322111311312232333323332131233232113212212322211211312

12223312211213331132122131213221311323333123312321221211122323211121213311213222

12211213331333331332232331331132221331133312122322111111233113323222322223332132

22111113113321323312313332222232123313221211222112233223222321121223223131331332

33221212121331121113121132221221123113123322112112233233213233222131311132133132

31131232131131112311231312233133232223131311311321313212223211333322221211222132

33233311313323211313212313113313322312222233132322323232233223112211331231133232

33211223331212121222323322213221121322332223311322133113331313231321133223333112

13311312231122123133121112221122211111223231133121122233312122312132113221323132

11233111111132223221211112233211312121333311223212222311313111221331121333332232

11313221311333322333323111322222222113112212131113331131313311122332132111331232

21321311131233121213122223211233113213211333321121122212131332332131133213233322

11332233121133313322333312313112311332231333213222122211332133212231221122223322

12121211113211231231133222332122213332322231213213133123222113122122233133111112

13331212323331223212113133212232211333321221311222313213223211133323323322232132

33333312121212321232312212331312122213223111132233213212122211113123223121311212

31233113333311223223113121111123232211312232212233332122312221333121212132312232

12221332331223332311131122333131232121221313321313211232122323111221223213113222

33123321223222133322323122121323121313132211232131321331332122112111112132232132

11222333313223123311113131113231311332113113221331133132122111333321222333321232

11232223222231112232213323211112332333232122111123123232123231133221212211112112

31233313111321113221323231133313122131212222113123122213113213333222313222233122

13333313332332323111223313133231222131313313121222112123331122332132211232311332

11131223221311132211113211321223231321231332131112312131223313211223112133333122

11111111111111111111111111111111111111111111111111111111111111111111111111111110

#end\_map

#flow

(11,2) (12,2) 5

(3,23) (2,23) 10

(25,10) (25,11) 15

(45,46) (45,47) 20

#end\_flow

#taxi

No.1 0 2 (1,1)

No.2 0 2 (2,0)

No.3 0 2 (3,3)

No.4 0 2 (4,4)

No.5 0 2 (5,5)

No.6 0 2 (6,6)

#end\_taxi

Consule输入样例：

请求：  
[CR,(48,8)(6,53)]

[CR,(16,56)(59,1)]

[CR,(27,77)(26,57)]

checkTaxi查询：

$5

$6

$7

道路通断：(这两条本来不一定是通的)

#(7,10) (8,10) 0

#(7,20) (8,20) 0