**设计思路概述**

本程序采用了多线程的设计。其中，输入类和调度类是线程。每辆出租车都是一个线程，能够根据状态独立地进行运动。但订单类不是线程。

首先从文件中读入地图文件并解析生成地图，然后利用BFS算法计算6400个点中每两个点间的最短路径（需要大概5秒）。从控制台读入字符串，解析并生成订单，加入订单序列。遍历订单序列，将抢单的出租车加入到订单的抢单序列中，3秒后从序列中选出一辆，修改其状态，使其按最短路径行驶完成订单。对于每一个订单都将操作输出到相应的文件。还可从控制台读入相应字符串来进行出租车状态和符合状态的出租车的查询。

本程序没有使用类的继承。

**各类及各方法的功能**

1 坐标类 Points.java

主要属性：点的序号，坐标，文件中的数值，上下左右是否有连线（boolean）

主要方法：Initial() 根据文件中的点的数值对点进行初始化，形成地图

2 地图类Map.java

主要属性：保存地图点的数组，每两点间最短路径距离的数组和方向的数组，

主要方法：Map ParseMap() 用于从文件中读地图

void BFS() 求一个点到其他所有点最短路径的方法

3 输入类 Input.java (extends Thread)

主要属性：订单序列

主要方法：run方法 解析控制台输入的三种输入（订单、查询出租车状态和查询符合状态的出租车）

4 出租车类 Taxi.java (extends Thread)

主要属性：序号，当前坐标，订单，状态，信用度

主要方法：void runRandom() 处于等待服务状态时的运行方法

void runShortest() 处于服务或接单状态时的运行方法

run方法 出租车作为线程的运动方法

5 订单类 Order.java

主要属性：发出时间，发出地点坐标，目标地点坐标，抢单的出租车序列，用于输出的文件

主要方法：Order ParseOrder() 用于解析从控制台读入的字符串并生成订单

void JudgeTaxi() 判断出租车是否抢单的方法

int ChooseTaxi 选择接单的车的方法

6 订单序列类 Orderlist.java 用于让输入类、调度类和出租车类共享订单序列

主要属性：Order的ArrayList

主要方法：添加、删除序列中订单的方法

7 调度类Dispatch.java (extends Thread)

主要属性：地图，所有出租车，订单序列

主要方法： run方法 遍历订单序列并分配给出租车

8 系统类 Uber.java

主要属性：开始时间

主要方法：main方法，初始化并开启所有线程

9 Gui接口类 gui.java 提供gui接口