# 各模块设计说明

### Mysql模块

**select City函数算法：**

根据传入的mysql语句，在数据库中进行查找，并返回QSqlQuery类型

**功能:**

向Widget主窗口传递数据

### **Map模块：**

**plane\_move,train\_move,car\_move函数算法：**

每种交通工具有不同的速度，根据传入的不同交通工具来调用不同函数进行路径绘制

### **Widget模块：**

**updateCurrentTime函数算法：**

每当计时器超时，时间向前移动6分钟，并更新文本框数据

**findWay函数算法：**

对所用到的数据进行初始化，并根据输入的策略选择不同函数，之后进行Makeplan制定路径

**Dijkstra函数算法：**

对于起始城市，设置其到达时间，与起始城距离，当前城市，是否经过城市的初始值，当当前城市不是目的城市且没有经过时，寻找距当前城市风险值最小的城市，并更新其到达时间和与起始城市的距离，并标记此城市已经经过，重复以上过程，直到到达目的城市。

**updateCity函数算法：**

在数据库中寻找从当前城市出发的时刻表，并一一计算其风险值，并更新plan每个城市的移动策略

**DFS函数算法：**

在数据库中寻找从当前城市出发的时刻表，先计算其总消耗时间，若时间限制没有超过，则进行下一步，否则直接结束。在时间限制没有超过的条件下，若该城市没有访问过且不是目的城市时，计算风险值，更新到达时间、tempPlan每个城市的移动策略，并标记该城市已经经过，进行进一步的递归；若在时间限制内且该城市是目的城市，计算风险值，更新到达时间表、tempPlan每个城市的移动策略，如果风险值比目前最小风险值还要小，则更新plan为tempPlan。

**Makeplan函数算法：**

根据目的城市和每个城市的移动策略，反向递归城市的路径，并将其压至path，mapPath栈中

**输出及绘图处理：**

将目的城市的到达时间输出，并按照栈中的顺序出栈，输出策略到解决方案文本框；当当前时间与解决方案中每个城市的出发时间相同时，向**mapWidget**窗口传递交通工具类型、出发城市、到达城市。