**数据结构课程设计**

计算机学院

班级 2014211306

学号 2014211283

姓名 史柠玮

班级 2014211306

学号 2014211295

姓名 韩瑷晶

* [设计任务的描述](#_设计任务的描述)
* [功能需求说明及分析](#_功能需求说明及分析)
* [总体方案设计说明](#_总体方案设计说明)
* [数据结构说明和数据字典](#_数据结构说明和数据字典)
* [各模块设计说明](#_各模块设计说明)
* [范例执行结果及测试情况说明](#_范例执行结果及测试情况说明)
* [评价和改进意见](#_评价和改进意见)
* [用户使用说明](#_用户使用说明)

# 设计任务的描述

城市之间有三种交通工具（汽车、火车和飞机）相连，某旅客于某一时刻向系统提出旅行要求，系统根据该旅客的要求为其设计一条旅行线路并输出；系统能查询当前时刻旅客所处的地点和状态（停留城市/所在交通工具）。

# 功能需求说明及分析

* 城市总数不少于10个（13个）
* 建立汽车、火车和飞机的时刻表（航班表）
  + 有沿途到站及票价信息
  + 不能太简单（不能总只是1班车次相连）
* 旅客的要求包括：起点、终点、途经某些城市和旅行策略
* 旅行策略有：
  + 最少费用策略：无时间限制，费用最少即可
  + 最少时间策略：无费用限制，时间最少即可
  + 限时最少费用策略：在规定的时间内所需费用最省

# 总体方案设计说明

软件开发环境、总体结构和模块划分等

Windows下开发，使用Qt Creator作为IDE，MySQL数据库进行时刻表调用，C++语言编程

目前为单一窗口，如果需要显示地图，可以增加一个窗口

模块：

Main：调用其他各个模块

Widget：主窗口，其上有输入信息和输出信息

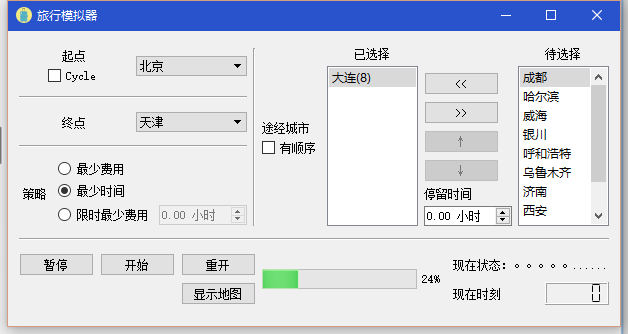
Route：地图窗口，可以显示地图以及当前的位置，预计路径行程等

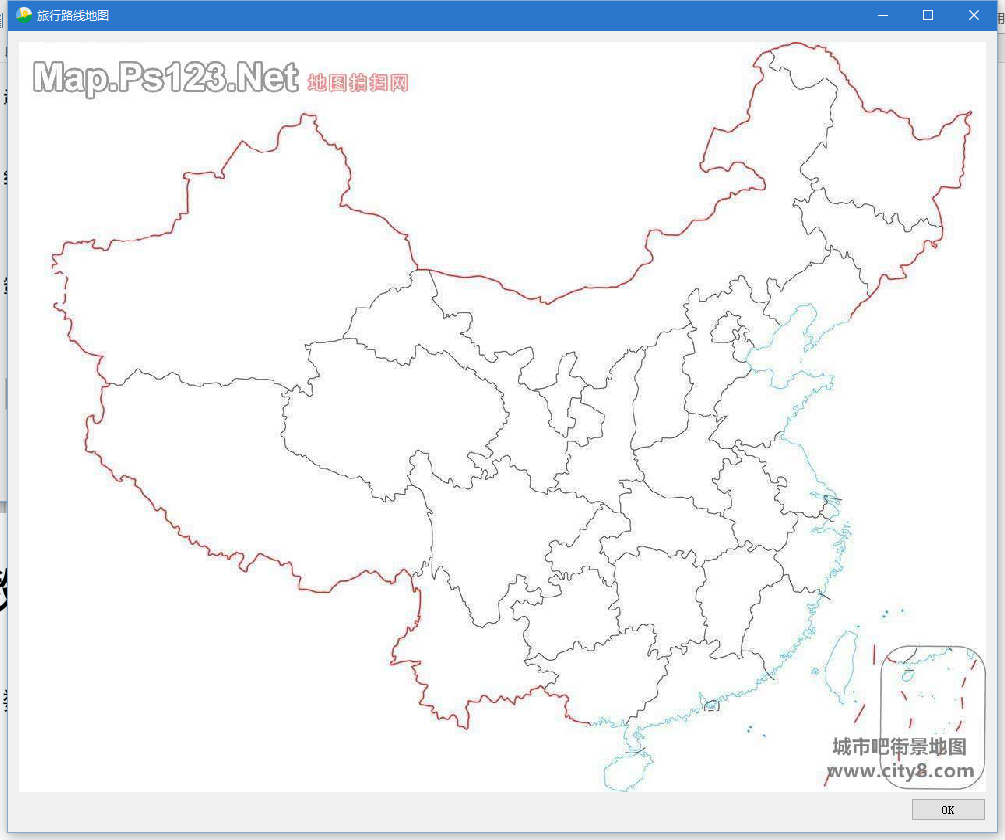
Passenger：储存输入的信息，当前状态，计算后的预计行程等

LogFile：记录日志

TimeTable：进行数据库的访问，将访问数据库封装成该类的方法。

//应该还有一个计时器来模拟时间流动





# **数据结构说明和数据字典**

（数据名称、用途等）

struct Status//保存当前状态

{

char **transport**;//当前交通工具

QString **curCity**;//当前城市

QString **nextMove**;//下一步

};

class Passenger//当前乘客的信息

{

public:

Passenger();

enum POLICY{minTime,minCost,timeLimitCost};

void setStart(QString s);

QString getStart();

void setEnd(QString e);

QString getEnd();

void setPolicy(int p);

int getPolicy();

void setLimitTime(double L);

double getLimitTime();

void setWayCities(QList<QPair<QString, double>> W);

QList<QPair<QString, double>> getWayCities();

void setSequence(bool checked);

bool isSequence();

private:

QString **start**;//起点

QString **end**;//终点

int **policy**;//策略

double **limitTime**;//限时最低价格的限制时间

QList<QPair<QString, double>> **wayCities**;//途经城市

bool **sequence**;//是否有顺序

Status **curStatus**;//暂时打算用作储存当前状态

};

struct EdgeType {

int **cost** = A\_BIG\_INT;

};

//邻接矩阵

class Graph //

{

public:

Graph();

void CreateGraph\_MinCost();//根据数据库建立最低价格的矩阵

int Dijkstra(QString start\_city, QString end\_city, vector<QString> &out); //迪杰斯特拉求最短路径

int LeastCost(QString start\_city, QString end\_city, vector<QString> &mid\_city, int isOrdered, vector<QString> &rout); //求总路径

private:

vector<QString> **vertex**;//城市名表

EdgeType **arc\_MinCost**[100][100];//邻接矩阵，可看作边表

int **numVertex**;//图中当前的顶点数

int **numEdge**; //图中当前的边数

};

struct Info//数据库中一条数据

{

QString **trainnumber**;

QString **departcity**;

QString **arrivecity**;

QTime **departtime**;

QTime **arrivetime**;

int **price**;

int **id**;

};

class TimeTable

{

public:

TimeTable();

~TimeTable();

static int getMinPrice(QString start, QString goal); //根据起点终点从数据库中寻找最少价格

static Info getInfo\_MinCost(QString start, QString goal); //根据起点终点从数据库中提取最少价格的那一条数据

private:

QSqlDatabase **db**;

static QMap<QString,QString> **full2Short**; //全名映射到简称

};

# 各模块设计说明

算法思想、算法、特点及与其它模块的关系等

在widget窗口中输入，即时设定好passenger的成员变量，点击开始后，执行算法。

最少费用：价格作为有向网的权值，利用迪杰斯特拉算法设计出路线。

# 范例执行结果及测试情况说明

# 评价和改进意见

# 用户使用说明