

《创新训练课B》课程设计报告

学院： 机电工程与自动化学院

题目： 模拟地铁自动售票系统

班级： 自动化2班

姓名： 黄继凡

学号： 200320224

教师： 吴晓军

上交日期： 2021.10.10

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 7.5 | V1.1 | 完成了基本的交互界面框架的构造、具备简单的计算票价的功能 | 黄继凡 |
| 7.8 | V1.2 | 对支付的流程、界面进行了完善，对于各个界面之间的切换进行了优化，避免闪屏的情况。对线路查询界面，购票界面的信息显示功能进行了完善、拓展。 | 黄继凡 |
| 7.10 | V1.3 | 增加了线路查询界面地图拖拽、放大缩小查看的功能 | 黄继凡 |
| 7.15 | V1.4 | 对程序中运用弗洛伊德算法求两点最短路径的相关代码进行了修正，可以正常输出从编号较大到编号较小的站点的路径 | 黄继凡 |

**目 录**

[1 引言 1](#_Toc519846913)

[1.1编写目的 1](#_Toc519846914)

[1.2背景 1](#_Toc519846915)

[1.3参考资料 1](#_Toc519846917)

[2 任务概述 1](#_Toc519846918)

[3 需求分析 1](#_Toc519846919)

[3.1 用户需求分析 1](#_Toc519846920)

[3.2 运行环境 1](#_Toc519846921)

[4 功能及操作介绍 1](#_Toc519846922)

[4.1 操作 1](#_Toc519846923)

[4.2 功能 1](#_Toc519846924)

[5 系统设计 2](#_Toc519846925)

[5.1 总体架构设计 2](#_Toc519846926)

[5.2 模块分析与设计 2](#_Toc519846927)

[5.3软件结构（流程图） 2](#_Toc519846928)

[6 调试与测试 2](#_Toc519846929)

[6.1调试过程 2](#_Toc519846930)

[6.2测试结果 2](#_Toc519846931)

[7 编程中遇到的问题 2](#_Toc519846932)

[7.1 问题1 2](#_Toc519846933)

[7.2 问题2 2](#_Toc519846934)

[7.3 问题3 2](#_Toc519846935)

[7.3 问题4 2](#_Toc519846935)

[7.3 问题5 2](#_Toc519846935)

[8分析总结与心得体会 2](#_Toc519846936)

# 1 引言

## 1.1编写目的

尝试模拟地铁购票过程中，自助购票机的交互界面，模拟乘客使用自助购票机时的流程。

## 1.2背景

当前各大城市人口数量的增加使得城市交通的需求达到了新的高度，地铁在各大城市中逐步成为城市内重要的出行方式，而地铁的售票系统也因此在地铁运营、管理方面有着相当程度的重要性。

## 1.4参考资料

[1]Savitch WalterJ. Problem Solving with C++[M]. 清华大学出版社, 2008.

[2]陆文周. Qt 5开发及实例[M]. 电子工业出版社, 2015.

[3]程杰. 大话数据结构[M]. 清华大学出版社, 2011.

# 2 任务概述

模拟地铁售票系统，需要模拟用户购买地铁票的基础流程，

# 3 需求分析

## 3.1 用户需求分析

用户购票时，可能由于初次使用，不了解使用方式，需要有一定的引导或说明来辅助操作。

购票时，用户可能对所要前往的地点的路线不了解，因此有必要设计线路查询的功能，用户可以使用此功能了解目的地的相关线路信息。

开始购票后，用户需要了解到达自己的目的起所需要的票价以付费，因此应当设置输入框等，获得用户的目的地，系统自动计算得出路线并根据计价规则给出票价，并可让用户设置需要购买的票数。

在此期间用户的误操作诸如错误输入等可以撤回，或不影响系统的正常工作，并可返回到上一级界面重新进行操作。

购票完成后，之后使用的用户可以继续购票。

## 3.2 运行环境

操作系统：windows 10

Qt版本：Qt 6.1.2

# 4 功能及操作介绍

## 4.1 操作

打开系统后，将会弹出用户须知界面，可以使用滚轮查看下方信息，点击“我已知悉”或点击右上角可关闭改该界面。

在系统界面，点击“退出系统”或右上角可以关闭系统。点击“线路查询”按钮可以进入线路查询界面，在右侧输入框可以输入要查询的站点，左侧地图可以点击“+”、“-”按键对地图进行放大、缩小的操作。

在线路查询界面或在系统主界面均有“开始购票”按钮，点击后进入购票界面，系统将会显示路线图，输入起点站后点击“确认”按钮进入终点站的选择。

在终点站选择界面，可以选择点击路线图下方的各个按钮显示各个路线的站点图，点击“恢复”按钮可以恢复为原先的总路线图。

输入终点站后，点击“购票”按钮，进入票数的选择，可在该界面输入要购买的票数。输入后点击“购买”按钮，进入模拟支付界面，输入支付金额，点击“确认支付”按钮，系统将会比较支付金额和票价，自动找零或显示支付金额不足。

以上各个界面均可点击“返回”或“取消”按钮回到上一级界面。

## 4.2 功能

1．购票：用户可点击“开始购票”按钮后，进入购票界面，之后根据提示输入起点站、终点站、票数等可进行支付购买。

2：路线显示：用户输入起点站和终点站后，系统会自动计算并显示用户的参考路线。

2.线路信息查询：用户点击“线路查询”按钮后，可进入线路查询界面，输入要查询的站点后，系统将会显示该站点所在线路以及相关运营信息。

4：地图查看：在线路查询界面，用户可放大并拖拽地图进行查看，方便确认站点信息。

# 5 系统设计

## 5.1 总体架构设计

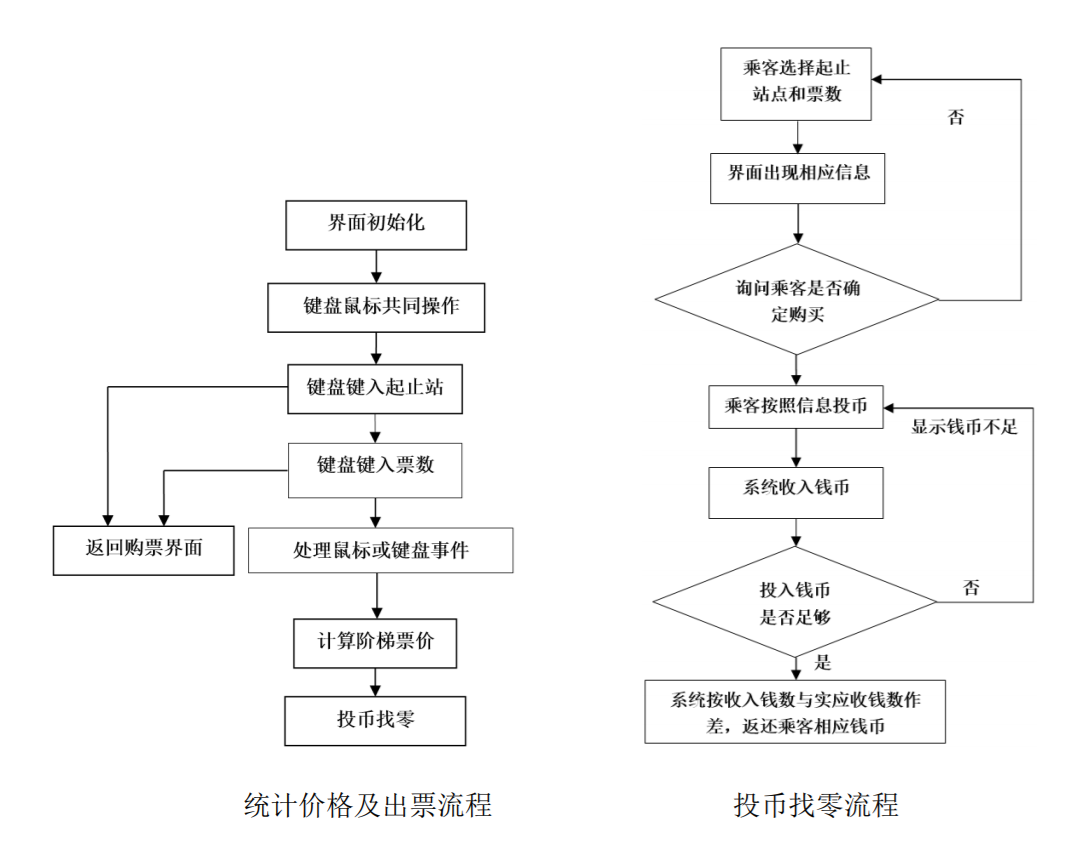
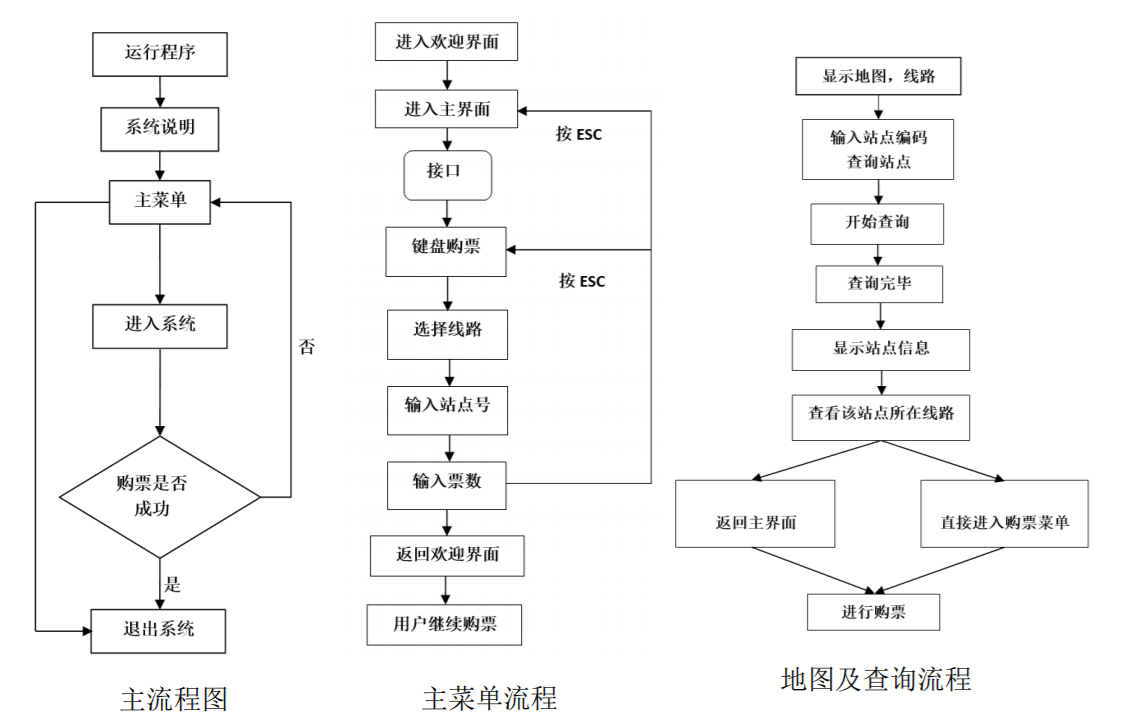
模拟地铁自动售票系统需要模拟用户使用地铁自主购票机的购票流程，应当具有站点查询，以及购票相关的起止站输入、计算票价、获得用户输入金额并进行判断等功能。

## 5.2 模块分析与设计

模拟地铁自动售票系统主要需要在图形交互界面方面进行较多的设计。根据各个界面之间的关系，可以大致的划分为以下几组界面。

1. 用户提示界面：此界面下显示系统操作流程和地铁乘客须知的信息，可以通过鼠标滑轮查看该界面的信息，点击按钮后可以关闭该界面。
2. 系统主界面：在该界面需要显示可使用该系统相关功能的按钮，包括购票、线路查询、退出按钮，并通过各个按钮联动各个界面的显示、隐藏。
3. 购票界面组：在购票功能相关界面，需要设置文本输入框，获得用户输入的站点、票数、付费金额等，并设置按钮确定用户的输入操作以进入下一级操作界面或者给出相关的错误提示。此外，在终点站输入的界面，设置按钮可以让用户通过按钮显示地铁线路各条路线，并在获得用户的起止站后，计算站点间的距离等，给出票价，以进行后续用户付费的流程。另外，需要设置键盘按键事件，按下Esc键后，返回上一级界面。
4. 线路查询界面：在该界面，需要设置文本输入框获得用户输入的站点，并设置文本显示控件，接受用户输入站点后，进行匹配，在匹配成功后显示站点相关信息。此外，设置了地图查看功能，用户可以点击放大、缩小按钮控制地图缩放比例，并可点击恢复按钮，恢复地图的默认显示，在地图的显示范围内，可以通过鼠标拖拽地图，进行详细的查看。
5. 地铁站点距离计算模块：除了上述的交互界面设计外，为使得购票功能完整，需要获得用户起点站、终点站之间的某种距离关系以计算票价，在此设计了一个文件用以计算站点之间的关系。系统运行前，在该模块相关文件中对线路站点信息进行初始化，从外部接受两个参数作为起点站和终点站，使用弗洛伊德算法获得两站点间的最短路径，返回两站地间的最短距离。

## 5.3软件结构（流程图）



# 6 调试与测试

## 6.1调试过程

在本次课程设计项目的程序编写中，遇到了不少的问题，也进行了不少的调式工作，而让我深刻的感受到调试的重要性的经历则是对地图拖拽以及缩放功能的实现。初次想要实现这个功能时，由于缺少对相关函数使用的经验，并没有完成。后续实现了一个基本的实验性图片拖拽功能，但在程序上使用时出现了拖拽后图片绘制位置莫名跳跃，拖拽时图片无法正常跟随鼠标，以及进行下一次拖拽操作时，图片位置跳跃的情况。

在调试过程中，需要使用Qt自身的qDebug函数输出与该功能实现相关的点的坐标。

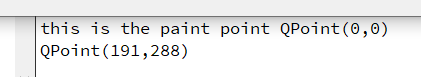


图6.1.1

Qt以窗口左上角为坐标系的原点，图中QPoint数据为鼠标点击时，鼠标相对于界面左上角的坐标。

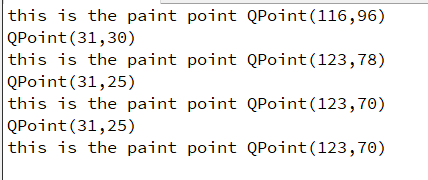


图6.1.2

该图中，QPoint数据为鼠标点击后，进行移动时相对于初次点击的点的坐标，this is the paint point 的数据为进行拖动时，要拖拽的图片的左上角即图片的绘制点相对于窗口原点的坐标。

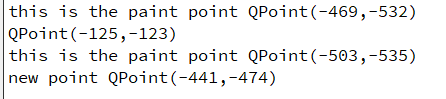


图6.1.3

该图中，new point的相关信息为鼠标松手后，程序所记录的图片绘制点的坐标，可以看到这里存在着一个数值上的跳动。

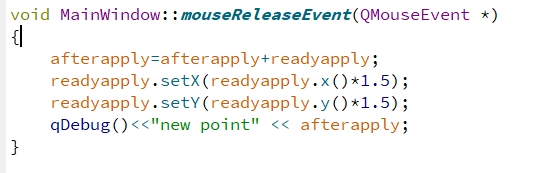


图6.1.4

经过检查，发现是在改变了图片的缩放比例后，没有对存储鼠标移动后，图片绘制点的改变进行合适的处理，在此处仍然使用了原始的鼠标相对于进行了点击的点移动的坐标数据，而没有考虑缩放倍数的问题，因此造成了数据跳动。

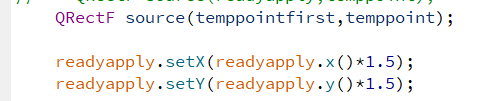


图6.1.5

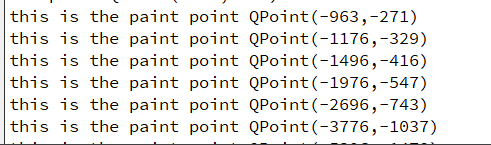


图6.1.6

另外，注意到原先编写程序时，直接将对缩放比例倍数处理的相关代码写在了绘图事件中，而绘图事件会在界面隐藏、打开后以及各类其他情况下被触发，那么在绘图事件相关代码中进行改变的数据将会多次异常执行改变的指令，最终导致功能无法正常实现，对相关的代码进行了处理后，解决了这一问题。

## 6.2测试结果

经过检查，未发现该功能现在存在影响正常使用的问题，可认为测试通过。

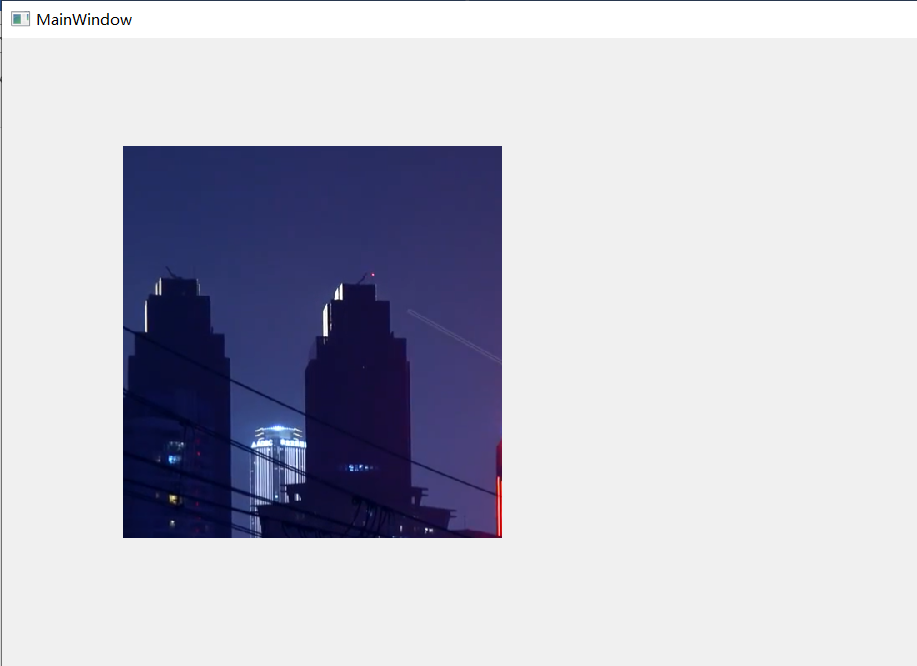


图6.2.1

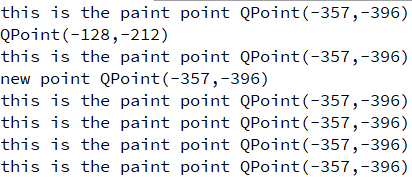


图6.2.2

以上两图为该试验性功能进行了相关修复后，在窗口界面的显示以及数据的输出，可以看到数据的跳变已经被修复，绘图事件的多次进行也没有使绘图点位置发生异常改变。

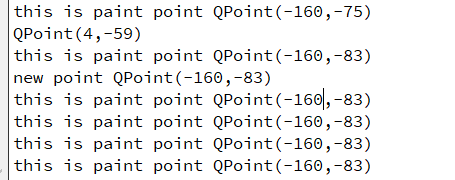


图6.2.3



图6.2.4

在运用到编写的系统程序上并进行相应后，未发现数据出现异常，功能的使用没有出现之前发现的问题，可以认为该功能基本实现。

相关实现代码：

QPoint readyapply;

QPoint prereadyapply;

QPoint afterapply=QPoint(0,0);

void map\_query::***mousePressEvent***(QMouseEvent \*e)

{

if(e->position().x()<550&&e->position().x()>150&&e->position().y()>180&&e->position().y()<580)

{

prereadyapply.setX(e->position().x());

prereadyapply.setY(e->position().y());

}

}

void map\_query::***mouseMoveEvent***(QMouseEvent \*e)

{

if(e->position().x()<550&&e->position().x()>150&&e->position().y()>180&&e->position().y()<580)

{

readyapply.setX(prereadyapply.x()-e->position().x());

readyapply.setY(prereadyapply.y()-e->position().y());

repaint();//与绘图事件联动，实现鼠标拖拽改变图片的显示情况

}

}

void map\_query::***mouseReleaseEvent***(QMouseEvent \*)

{

readyapply.setX(readyapply.x()\*window\_length/400.0);

readyapply.setY(readyapply.y()\*window\_length/400.0);

afterapply=afterapply+readyapply;

readyapply=QPoint(0,0);

}

# 7 编程中遇到的问题

## 7.1 问题1

使用全局变量时，由于声明或定义有误而报错。通过查阅相关资料后，了解到需要在文件头文件使用extern声明为全局变量，在执行文件进行定义，并要在需要使用该变量的文件中包含相关的头文件来使用。

## 7.2 问题2

使用Qt绘图事件时，由于绘图事件会因多种因素而触发，如果全局变量或静态局部变量在这部分代码中使用，可能会造成数据异常，影响界面显示。通过对相关变量的合适处理，避免了绘图事件多次触发对数据的异常影响。

## 7.3 问题3

编写显示用户起点站到终点站的路线的相关代码中，发现经常会出现显示的文本有误的情况。经过检查发现是保存线路站点的数组重置时机设置有误，在更改输入的站点后，由于没有合理重置用于输出显示的数组，导致文本接受了应当清空的站点名，使得显示的路线有误。在对数组进行了合适的重置后，这个问题得以解决。

## 7.4 问题4

用于获得用户输入的起止站的最短距离以及路径的算法有误，在寻找从编号较大到编号较小的站点的路线时，无法正常的输出路线上的所有站点，出现信息输出不全的情况。经过进一步的学习、检查，发现这是由于进行递归输出途径站点时，递归条件存在问题，只能完整输出从编号较小的站点到编号较大的站点的信息。对递归条件以及相关的存储数据进行调整后，可以完整输出两站点最短路径上的其余站点。

## 7.5 问题5

在进行线路查询界面的线路图拖拽功能的实现时，发现经常会出现图片绘制位置突然改变，拖动时鼠标点击到的图片上一点不能正常跟随鼠标，以及拖拽完成后，进行一下次拖拽时图片位置跳跃的情况。经过对所使用的函数以及设置存储图片绘制的位置的变量进行检查后、输出调试后，发现绘图事件会在多种情况下触发，导致存储图片绘制位置的变量值出现异常，修改相关变量值改变的条件后，解决了图片绘制位置无鼠标操作情况下突然变化的问题，根据图片缩放比例调整图片绘制点的位移比例后，解决了图片不能正常跟随鼠标的问题。

# 8分析总结与心得体会

本次创新训练课的题目对我来说是个不小的挑战，由于C++和图形界面编程等对我来说都是几乎没有接触过的内容。不过由于时间上较为充裕，并且选题的技术要求等比较容易理解，在一段时间的学习后，还是掌握了一些这方面的知识，有一些收获。

在这次课程的作业上，由于开始对地铁站站点距离计算没有较清晰的认知，尝试了几类方法来计算距离，但只能处理简单情况，对于多线路交汇的较复杂情况，由于编写代码太过繁琐且无法较好的处理多线路交汇于一点等情况而最终都没有应用到正式程序上。最终经过一定的学习了解，认为将站点抽象成图进行处理是一个比较好的解决方式，这让我感受到对问题要有着明确认知，这是能够解决问题的重要条件。

在完成课程题目的过程中，我翻阅了关于C++、Qt编程以及数据结构的相关书籍，尽管只做到了较简单的应用，但对完成这次的题目起到了较大的帮助作用。另外也从一些技术社区的交流中，进一步了解到了相关的知识，这让我感受到交流在解决问题方面的重要性，在完成了基本功能的实现后，我也在社区上分享了关于一个功能实现的具体细节。