**交易平台****系统**

**设计文档**

版本：2.0

数据结构一班六组

21/01/2018

版本修订历史

| 版本号 | 日期 | 编写人员 | 评审人员 | 核准者 | 影响模块及变更概要 |
| --- | --- | --- | --- | --- | --- |
| V1.0 | 18/01/2018 | 崔辉、王姗姗、李晓龙、孙昊、许荣耀 | 崔辉 | 崔辉 | 物流信息模块添加最短路径以及最最少花费路径的输出 |
| V2.0 | 19/01/2018 | 崔辉、王姗姗、李晓龙、孙昊、许荣耀 | 崔辉 | 崔辉 | 算法模块，查询改进为哈希算法，提高查询效率 |
| V3.0 | 20/01/2018 | 崔辉、王姗姗、李晓龙、孙昊、许荣耀 | 崔辉 | 崔辉 | 将树视图改为多页文本视图，提高载入效率 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[**交易平台** 1](#_Toc485807763)

[**库存信息同步系统** 1](#_Toc485807764)

[**设计文档** 1](#_Toc485807765)

[**1 概述** 4](#_Toc485807766)

[**1.1 开发背景** 4](#_Toc485807767)

[**1.2 开发目标** 4](#_Toc485807768)

[**2 总体设计** 4](#_Toc485807769)

[**2.1 简述** 4](#_Toc485807770)

[**2.2 系统框架** 4](#_Toc485807771)

[**2.3 运行环境** 4](#_Toc485807772)

[**3 详细设计** 4](#_Toc485807773)

[**3.1 程序流程图** 4](#_Toc485807774)

[**3.2 算法设计** 5](#_Toc485807775)

[**3.2.1查询算法** 5](#_Toc485807776)

[**3.2.2排序算法** 5](#_Toc485807777)

[**3.2.3最优路径算法** 5](#_Toc485807778)

[**3.3 存储分配设计** 5](#_Toc485807779)

[**3.3.1商品信息存储结构** 5](#_Toc485807780)

[**3.3.2物流信息存储结构** 6](#_Toc485807781)

[**3.4 接口设计** 6](#_Toc485807782)

[**3.4.1 用户界面接口** 6](#_Toc485807783)

[**3.4.2 商品库货信息导入模块接口** 9](#_Toc485807784)

[**3.4.3 商品筛选模块接口** 10](#_Toc485807785)

[**3.4.4价格排序模块接口** 10](#_Toc485807786)

[**3.4.5商品信息更新模块接口** 10](#_Toc485807787)

[**3.4.6最佳物流路径推荐模块接口** 10](#_Toc485807788)

[**3.5 注释设计** 10](#_Toc485807789)

**1 概述**

**1.1 开发背景**

对于交易平台来说，每时每刻都要对客户的需求进行及时的响应，当数据需求量很大时，人工很难对客户进行及时的响应工作，这时交易平台就要起到当客户输入请求时能够及时的告知客户所需的信息的作用。

**1.2 开发目标**

交易平台系统的功能需要包括载入信息，商品的上下架，物流信息的查询，商品信息的查询，商品的添加，修改以及删除等功能

**2 总体设计**

**2.1 简述**

系统采用C结构模式。系统的分析设计采用面向过程的技术。

**2.2 系统框架**

系统采用MVC框架。

**2.3 运行环境**

操作系统：Linux

运行环境：CentOS6.0（32b）或CentOS6.5（64b）平台

编程语言：C

界面显示：GTK+2.0

**3 详细设计**

**3.1 程序流程图**

登录界面

交易平台

修改信息

按价格排序

商品搜索

帮助界面

商品下架

商品上架

查看物流

商品载入

最少花费

最短时间

批量载入

库存载入

功能关键字

价格

形状

存货

卖家地址

名称

品牌

材质

颜色

**3.2 算法设计**

**3.2.1查询算法**

哈希：利用哈希的思想，利用搜索的关键字建立指针数组来做索引，指针数组指向的就是各个链表的头结点，一个结点后面跟着一个链来记录具有同一关键字的元素。查询时，只需要根据关键字找到对应的索引，将对应的链输出即可；

**3.2.2排序算法**

快速排序：快速排序是基于一次划分，就是进行多次一次划分，一次划分可以依据枢轴元素将元素分为大于和小于该元素的两类元素，所以在递归的进行多次一次划分之后就可以将一组数据变为有序；

**3.2.3最优路径算法**

迪杰斯特拉：求图中两点之间的最短路径有两种算法，迪杰斯特拉算法和弗洛伊德算法，相比来讲，迪杰斯特拉算法便于记录路径。

**3.3 存储分配设计**

**3.3.1商品信息存储结构**

链式存储结构（双向链表）

struct info

{

char name[22]; //名称

char city[15]; //城市

int restnum; //存货量

int price; //价格

char pinpai[15]; //品牌

char caizhi[15]; //材质

char shape[15]; //款式形状

char color[20]; //颜色

char use[50]; //功能描述

};

typedef struct node

{

struct info obj;

struct node \*next,\*pre;

}\*link;

**3.3.2物流信息存储结构**

顺序存储结构（邻接矩阵）

double map\_cost[17][17]={{0, 7, 10, 13, 10, 5, 12, 15, 9999, 7, 12, 9999, 12, 9999, 7, 9999, 9999}, {7,0, 3, 5, 4, 11, 9999, 9999, 9999, 15, 9999, 9999, 9999, 9999, 14, 9999, 9999},{10, 3, 0, 2, 7, 14, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,9999}, {13, 5, 2, 0, 9, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999}, {10, 4, 7, 9, 0, 15, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999}, {5, 11, 14, 9999, 15, 0, 7, 10, 12, 7, 12, 9999, 12, 9999, 12, 9999, 9999}, {12, 9999, 9999, 9999, 9999, 7, 0, 3, 5, 7, 12, 9999, 12, 9999, 9999, 9999, 9999}, {15, 9999, 9999, 9999, 9999, 10, 3, 0, 2, 10, 14, 9999, 15, 9999, 9999, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 12, 5, 2, 0, 12, 12, 9999, 9999, 9999, 9999, 9999, 9999}, {7, 15, 9999, 9999, 9999, 7, 7, 10, 12, 0, 5, 9999, 5, 12, 14, 9999, 9999}, {12, 9999, 9999, 9999, 9999, 12, 12, 14, 12, 5, 0, 11, 10, 9999, 9999, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 11, 0, 9999, 9999, 9999, 9999, 9999}, {12, 9999, 9999, 9999, 9999, 12, 12, 15, 9999, 5, 10, 9999, 0, 7, 13, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 12, 9999, 9999, 7, 0, 11, 19, 21}, {7, 14, 9999, 9999, 9999, 12, 9999, 9999, 9999, 14, 9999, 9999, 13, 11, 0, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 19, 9999, 0, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 21, 9999, 9999, 0}};

double map\_time[17][17]={{0, 0.7, 1, 1.3, 1, 0.45, 0.5, 0.5, 9999, 0.5, 1, 9999, 1.1, 9999, 0.5, 9999, 9999}, {0.7, 0, 0.5, 0.5, 0.5, 1.2, 9999, 9999, 9999, 2, 9999, 9999, 9999, 9999, 1.5, 9999, 9999},{1, 0.5, 0, 0.5, 0.5, 2, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999,9999}, {1.3, 0.5, 0.5, 0, 1.5, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999}, {1, 0.5, 0.5, 1.5, 0, 2, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999}, {0.45, 1.2, 2, 9999, 2, 0, 0.5, 1.1, 1.5, 0.5, 1.2, 9999, 1.5, 9999, 2, 9999, 9999}, {0.5, 9999, 9999, 9999, 9999, 0.5, 0, 0.5, 0.5, 1, 1.5, 9999, 1.5, 9999, 9999, 9999, 9999}, {0.5, 9999, 9999, 9999, 9999, 1.1, 0.5, 0, 0.3, 1.2, 1.5, 9999, 2, 9999, 9999, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 1.5, 0.5, 0.3, 0, 1.5, 1.5, 9999, 9999, 9999, 9999, 9999, 9999}, {0.5, 2, 9999, 9999, 9999, 0.5, 1, 1.2, 1.5, 0, 0.5, 9999, 0.5, 1.5, 2, 9999, 9999}, {1, 9999, 9999, 9999, 9999, 1.2, 1.5, 1.5, 1.5, 0.5, 0, 1.5, 1, 9999, 9999, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 1.5, 0, 9999, 9999, 9999, 9999, 9999}, {1.1, 9999, 9999, 9999, 9999, 1.5, 1.5, 2, 9999, 0.5, 1, 9999, 0, 0.5, 2, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 1.5, 9999, 9999, 0.5, 0, 2, 2, 3}, {0.5, 1.5, 9999, 9999, 9999, 2, 9999, 9999, 9999, 2, 9999, 9999, 2, 2, 0, 9999, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 2, 9999, 0, 9999}, {9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 3, 9999, 9999, 0}};

**3.4 接口设计**

**3.4.1 用户界面接口**

借助Glade进行界面设计，系统主界面设计如下：

登录界面设计如下：

. 

登录界面主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| window\_login | 登录窗口 | 无 |
| entry\_id | 用户名输入框 | 无 |
| entry\_password | 密码输入框 | 无 |
| button\_login\_ | 登录按钮 | on\_button\_login\_clicked |
| button\_login\_cancel | 取消按钮 | on\_button\_login\_cancel\_clicked |
| window2\_login | 错误信息提示界面 | 无 |
| button\_back | 关闭错误信息界面按钮 | on\_button\_back\_clicked |



系统主界面中主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| window\_main | 系统主窗口 | 无 |
| treeview\_goods | 商品显示界面 | on\_treeview\_goods\_cursor\_changed |
| entry\_search | 查找商品输入框 | 无 |
| checkbutton | 在当前页面查询 | on\_checkbutton1\_toggled |
| button\_change | 修改商品信息按钮 | on\_button\_change\_clicked |
| button\_load | 载入商品信息 | on\_button\_load\_clicked |
| button\_sort | 依据商品价格排序 | on\_button\_sort\_clicked |
| button\_wuliu | 物流查询 | on\_button\_wuliu\_clicked |
| button\_help | 帮助界面 | on\_button\_help\_clicked |
| button\_pre | 上翻页 | on\_button\_pre\_clicked |
| button\_next | 下翻页 | on\_button\_next\_clicked |
| button\_next\_sort | 排序页面下翻页 | on\_button\_next\_sort\_clicked |
| button\_pre\_sort | 排序页面上翻页 | on\_button\_pre\_sort\_clicked |
| button\_pre\_search | 搜索页面上翻页 | on\_button\_pre\_search\_clicked |
| button\_next\_search | 搜索页面下翻页 | on\_button\_next\_search\_clicked |

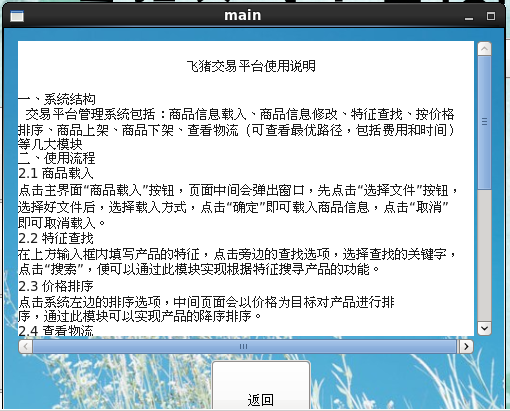
库货信息载入界面设计如下：



库货信息载入界面主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| window\_load | 商品载入界面 | 无 |
| entry\_loade\_show | 查找信息输入框 | 无 |
| button\_file\_choose | 文件选择按钮 | on\_button\_file\_choose\_clicked |
| combobox\_load | 载入方式复选框 | 无 |
| button\_load\_cancel | 取消按钮 | on\_button\_load\_cancel\_clicked |
| button\_load\_sure | 确定载入按钮 | on\_button\_load\_sure\_clicked |
| window\_file | 文件载入界面 | 无 |
| button\_file\_sure | 确定文件载入按钮 | on\_button\_file\_sure\_clicked |
| button\_file\_cancel | 取消文件载入按钮 | on\_button\_file\_cancel\_clicked |

帮助界面设计如下：



帮助界面主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| window\_help | 帮助界面 | 无 |
| button\_back | 返回按钮 | on\_button\_back\_clicked |
| textview\_help | 帮助文档 | 无 |

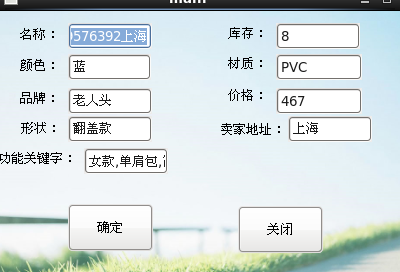
商品上架界面界面设计如下：



商品上架界面主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| Window1\_add | 商品上架界面 | 无 |
| entry\_name\_add | 商品名称输入框 | 无 |
| entry\_address\_add | 商品地点 | 无 |
| entry\_rest\_add | 剩余存货量 | 无 |
| entry\_price\_add | 商品价格 | 无 |
| entry\_sign\_add | 品牌 | 无 |
| entry\_material\_add | 材料 | 无 |
| entry\_shape\_add | 形状 | 无 |
| entry\_color\_add | 颜色 | 无 |
| button\_add\_sure | 确认添加 | on\_button\_add\_sure\_clicked |
| button\_add\_cancel | 取消添加 | on\_button\_add\_cancel\_clicked |
| entry\_describe | 功能描述 | 无 |
| Window2\_add | 错误信息提示界面 | 无 |
| Button\_return\_add | 错误信息提示界面关闭按钮 | on\_button\_return\_add\_clicked |

信息修改界面设计如下：



信息修改界面主要构件及响应函数设计如下：

|  |  |  |
| --- | --- | --- |
| **构件名称** | **构件描述** | **响应函数** |
| window\_change | 修改商品信息 | 无 |
| entry\_color\_change | 商品颜色文本框 | on\_checkbuttonshuxing\_toggled |
| entry\_singal\_change | 商品品牌文本框 | 无 |
| entry\_body\_change | 商品形状文本框 | on\_button\_change\_sure\_clicked |
| entry\_fountion\_change | 商品功能关键字文本框 | on\_button\_change\_cancel\_clicked |
| entry\_rest\_change | 商品存货文本框 | 无 |
| entry\_material\_change | 商品材质文本框 | 无 |
| entry\_price\_change | 商品价格文本框 | 无 |
| button\_change\_sure | 确认修改按钮 | on\_button\_change\_sure\_clicked |
| button\_change\_close | 取消修改按钮 | on\_button\_change\_close\_clicked |
| window1\_change | 信息提示界面 | 无 |
| button1\_change\_close | 信息提示界面关闭按钮 | on\_button1\_change\_close\_clicked |
| window2\_change | 错误信息提示界面 | 无 |
| button2\_change\_close | 错误信息提示关闭按钮 | on\_button2\_change\_close\_clicked |

物流界面设计如下：



物流界面主要构件及响应函数设计如下：

|  |  |  |  |
| --- | --- | --- | --- |
| **构件名称** | | **构件描述** | **响应函数** |
| window\_wuliu | 物流界面 | 无 |
| combobox1\_wuliu | 卖家地址 | 无 |
| Combobox2\_wuliu | 买家地址 | 无 |
| button\_wuliu\_sure | 确定查看物流按钮 | on\_button\_wuliu\_sure\_clicked |
| button\_wuliu\_cancel | 取消查看物流按钮 | on\_button\_wuliu\_cancel\_clicked |
| window1\_wuliu | 错误信息提示界面 | 无 |
| window2\_wuliu | 错误信息提示界面 | 无 |
| button1\_wuliu\_close | 关闭错误信息界面按钮 | on\_button1\_wuliu\_close\_clicked |
| eutton2\_wuliu\_close | 关闭错误信息界面按钮 | on\_button2\_wuliu\_close\_clicked |
| window3\_wuliu | 物流信息显示界面 | 无 |
| entry\_short | 最短时间路径文本框 | 无 |
| entry\_short\_time | 最短时间文本框 | 无 |
| entry\_cheap | 最少花费路径文本框 | 无 |
| entry\_cheap\_cost | 最少花费文本框 | 无 |
| button3\_wuliu\_close | 物流信息关闭按钮 | on\_button3\_wuliu\_close\_clicked |

**3.4.2 商品库货信息导入模块接口**

Loadcont.c：

1、商品信息加载按钮响应函数，实现弹出库货信息载入界面功能。

void on\_button\_file\_choose\_clicked(GtkObject\* widget,gpointer user\_date);

2、商品信息加载界面关闭按钮响应函数，实现关闭库货信息再入界面

.. void on\_button\_load\_cancel\_clicked(GtkObject\* widget,gpointer user\_date);

3、商品信息加载确定按钮响应函数，实现确定载入库货信息

void on\_button\_load\_sure\_clicked(GtkObject\* widget,gpointer user\_date);

4、文件载入取消按纽响应函数，实现关闭文件载入界面

void on\_button\_file\_cancel\_clicked(GtkObject\* widget,gpointer user\_date);

5、文件载入确定按钮响应函数，实现确定文件载入

void on\_button\_file\_sure\_clicked(GtkObject\* widget,gpointer user\_date);

**3.4.3 登录界面模块接口**

Logincont.c:

1、登录按钮响应函数，实现登录系统进入主界面

void on\_button\_login\_clicked(GtkObject\* widget,gpointer user\_data);

2、登录返回按钮响应函数，实现关闭提示错误界面

void on\_button\_back\_clicked(GtkObject\* widget,gpointer user\_data);

3、登录取消按钮响应函数，实现取消登录系统

void on\_button\_login\_cancel\_clicked(GtkObject\* widget,gpointer user\_data);

**3.4.4 主界面模块接口**

1、帮助信息按钮响应函数，实现弹出帮助界面

void on\_button\_help\_clicked(GtkObject\* widget,gpointer user\_date);

2、商品下架按钮响应函数，实现商品信息的删除

void on\_button\_decrease\_clicked(GtkObject\* widget,gpointer user\_date);

3、商品上架按钮响应函数， 实现弹出商品信息上架界面

void on\_button\_add\_clicked(GtkObject\* widget,gpointer user\_date);

4、物流信息查看按钮响应函数， 实现弹出物流信息界面

void on\_button\_wuliu\_clicked(GtkObject\* widget,gpointer user\_date);

5、商品信息排序按钮响应函数，实现商品按照一定的顺序进行显示

void on\_button\_sort\_clicked(GtkObject\* widget,gpointer user\_date);

6、商品信息修改按钮响应函数，实现弹出商品信息的修改界面

void on\_button\_change\_clicked(GtkObject\* widget,gpointer user\_date);

7、商品信息加载按钮相应函数，实现弹出商品信息加载界面

void on\_button\_load\_clicked(GtkObject\* widget,gpointer user\_date);

8、商品信息查询按钮响应函数，实现商品信息的相应显示

void on\_button\_search\_clicked(GtkObject\* widget,gpointer user\_date);

9、错误提示关闭按钮响应函数，实现关闭错误信息提示界面

void on\_button2\_change\_close\_clicked(GtkObject\* widget,gpointer user\_data);

**3.4.5 物流界面模块接口**

1、确定物流查看按钮响应函数，实现弹出物流信息界面

void on\_button\_wuliu\_sure\_clicked(GtkObject\* widget,gpointer user\_data);

2、关闭物流查看按钮响应函数，实现关闭物流查看界面

void on\_button\_wuliu\_cancel\_clicked(GtkObject\* widget,gpointer user\_data);

3、关闭物流信息按钮响应函数，实现关闭物流信息界面

void on\_button1\_wuliu\_close\_clicked(GtkObject\* widget,gpointer user\_data);

4、错误提示关闭按钮响应函数，实现关闭错误提示界面

void on\_button2\_wuliu\_close\_clicked(GtkObject\* widget,gpointer user\_data);

5、错误提示关闭按钮响应函数，实现关闭错误提示界面

void on\_button3\_wuliu\_close\_clicked(GtkObject\* widget,gpointer user\_data);

**3.4.6商品上架界面模块接口**

1、确定商品添加按钮响应函数，实现商品的添加

void on\_button\_sure\_add\_clicked(GtkObject\* widget,gpointer user\_data);

2、取消商品添加按钮响应函数，实现关闭商品添加界面

void on\_button\_cancel\_add\_clicked(GtkObject\* widget,gpointer user\_data);

3、错误提示关闭按钮响应函数，实现关闭错误提示界面

void on\_button\_return\_add\_clicked(GtkObject\* widget,gpointer user\_data)；

**3.4.7商品信息修改界面模块接口**

1、实现从视图中读到商品信息

void on\_treeview\_goods\_cursor\_changed(GtkObject\*widget,gpointer user\_data);

2、确认修改商品信息按钮响应函数，实现商品信息的修改

void on\_button\_change\_sure\_clicked(GtkObject\* widget,gpointer user\_data);

3、取消修改商品信息按钮响应函数，实现关闭商品信息修改界面

void on\_button\_change\_close\_clicked(GtkObject\* widget,gpointer user\_data);

4、错误提示关闭按钮响应函数，实现关闭错误提示界面

void on\_button1\_change\_close\_clicked(GtkObject\* widget,gpointer user\_data);

**3.4.8算法模块接口**

1、交换两个结点的值以及对应的指针指向关系

void swap(int \*a,int \*b);

2、一次划分，依据枢轴元素将数据分为大于该元素的和小于该元素的两类，一次划分就是进行快速排序的一个依据。

struct node \*partion(struct node \*pBegin,struct node \*pEnd);

3、快速排序，递归的进行一次划分，当头尾的指针相遇时排序也就完成。

void quick\_sort(struct node \*pBegin,struct node \*pEnd);

struct node\* sortList(struct node\* head);

4、字符串与数字的转换，为了建立索引，将对应的关键字转化为对应的数字，可以方便的找到对应的头指针

int wuliu\_change1(const gchar \*str);

int color\_change(const gchar \*str);

int caizhi\_change(const gchar \*str);

int pinpai\_change(const gchar \*str);

5、迪杰斯特拉,利用迪杰斯特拉算法依据权值求出最短的时间花费和运费花费

void mini\_time(int s, int e);

void mini\_cost(int s, int e);

6、建链表，在读入数据的时候，建立一个双向链表作为主链，之后的建立哈希就是 依据该链表。

void inithead();

7、创建哈希链表，依据要搜索的四个关键字建立相应的哈希表

void create\_city\_list(struct info a);

void create\_color\_list(struct info a);

void create\_caizhi\_list(struct info a);

void create\_pinpai\_list(struct info a);

**3.5 注释设计**

1、对文档的注释，设计如下：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright: SDUT 2018-2068

Author: 王姗姗

Data: 2018-01-20

Description:实现物流信息的查询

Emile:17853312495@163.com

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

2、对函数的注释，设计如下：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function: 实现最短路径的算法

in：int s 卖家地址

int e 买家地址

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

3、对变量的注释，设计如下：

GtkEntry\* entry\_update; //更新内容录入框

4、模块首部的注释，设计如下：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*系统主窗口构件指针\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GtkWindow\* window\_main;//主窗口

GtkEntry\* entry\_search;//查找内容

GtkComboBox\* combobox\_keywords;//商品关键词组合框