

高级语言程序设计

课程设计报告

|  |  |  |
| --- | --- | --- |
| 课程设计成绩（总分20） | | 分数 |
| 程序 | 程序设计完整，功能齐全（8-10分） |  |
| 程序设计基本完整，功能基本完成（0-7分） |
| 报告 | 结构清晰完整，报告格式正确，语言通顺（8-10分） |  |
| 格式不正确，封面或字体有误，语言欠通顺（0-7分） |
|  | 总分 |  |

|  |  |
| --- | --- |
| 课程名称 | 高级语言程序设计 |
| 课程编号 | 08306135 |
| 报告题目 | 网购系统 |
| 序 号 | 48 |
| 姓 名 | 王佳信 |
| 学 号 | 21122915 |
| 所在学院 | 计算机工程与科学学院 |
| 专业名称 | 智能科学与技术 |

2021年11月

目 录

[1. 数据格式与文件描述 5](#_Toc87307010)

[1.1 数据文件格式 5](#_Toc87307011)

[**2.** **问题分析** 7](#_Toc87307012)

[**3.** **总体设计** 8](#_Toc87307013)

[3.1 功能模块设计 8](#_Toc87307014)

[3.2 系统界面设计 8](#_Toc87307015)

[3.3 数据结构设计 9](#_Toc87307016)

[3.4 函数设计 12](#_Toc87307017)

[**4.** **详细设计** 14](#_Toc87307034)

[4.1 各个函数的调用关系图 14](#_Toc87307035)

[4.2 各个函数的功能描述 14](#_Toc87307037)

[**5.** **功能测试** 18](#_Toc87307052)

[6. 总 结 24](#_Toc87307065)

[7. 参考资料 24](#_Toc87307066)

网购系统

网购系统旨在给广大剁手党提供一个网上购物的平台，系统用户可以选择不同的身份执行不同的操作。如果是买家身份，浏览商品、购买商品的功能，如果是卖家身份，可以发布商品、修改商品、删除商品。如果是管理员身份，可以删除买家用户、卖家用户。

本实验要求设计一个简单的网购系统，涉及函数、结构体、链表、文件等方面的知识，学习利用链表处理数据的方法，熟练掌握文件操作，构建综合程序设计的思路及框架，提高综合设计软件系统的能力。

1. 数据格式与文件描述
   1. 数据文件格式

数据文件分为3个，买家信息、卖家信息、商品信息，每个文件中有多条数据。买家信息文件包括了此买家编号、姓名、购买商品的编号(一个或者多个，多个以逗号分割)。格式如图1-1所示。卖家信息文件包括卖家编号、姓名、联系方式、售卖的商品编号(一个或者多个，多个以逗号分割)，具体格式如图1-2所示。商品信息包括：商品编号、商品名称、商品描述、商品价格、商品件数、售卖商家编号(就一个)。具体格式如图1-3所示。

具体的买家信息、卖家信息和商品内容可自行设定。

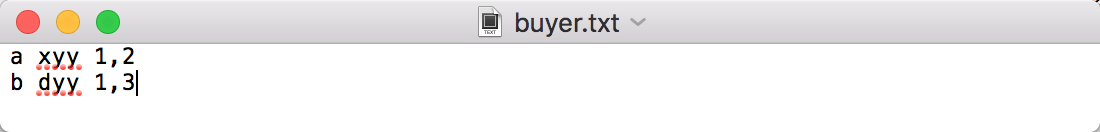


图1-1 买家信息

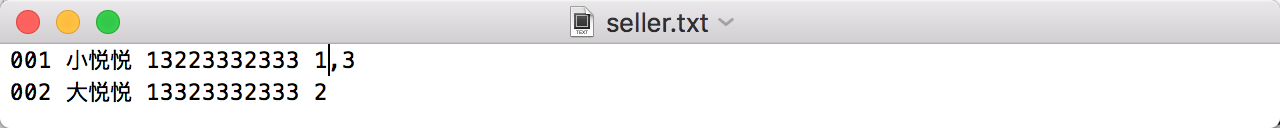


图1-2 卖家信息

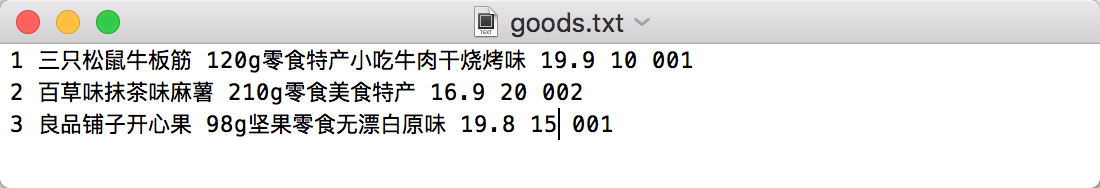


图1-3商品信息

* 1. 功能描述

设计一个简易的网购系统，利用单链表来处理买家信息、卖家信息与商品信息，要求实现如下功能。

1. 初始化菜单

程序开始运行后显示菜单，要求菜单能完全展示本系统的功能：提示三种用户身份(买家、卖家、管理员),并输出每项身份的功能权限。尽力做到设计美观。

1. 选择用户身份权限

考虑到本系统有用户身份，需要提示用户输入来选择此时的身份是买家、卖家、还是管理员。

1. 买家查询商品

选择买家身份。查询需要购买的商品名称或者店铺名称，如果相应商品有库存，输出满足要求的商品信息。如果没有此商品或者库存为0，输出提示信息。

可输出全部商品信息，若当前没有商品，输出提示信息。

1. 买家购买商品

选择买家身份。买家首先输入自己的编号，而后输入要购买的商品，如果没有库存，提示请更换商品，如果有库存，买家信息中购买商品处增加此商品编号，商品文件中，商品库存减小一份。

1. 卖家发布商品

选择卖家身份。卖家首先输入自己的卖家编号，买家用户输入待增加的商品的编号，如果编号存在，则判断商品是否是该卖家所售，如果是，修改库存，如果不是，输出提示信息，如果编号不存在，逐项录入新增商品信息。

1. 卖家修改商品

选择卖家身份。卖家首先输入自己的卖家编号，输入自己要修改商品编号，若商品不存在或不是该买家所售卖，输出提示信息。如果商品存在，则提示选择要修改的内容，并输出修改后的此商品信息。

1. 卖家删除商品

选择卖家身份。卖家首先输入自己的卖家编号，输入自己要删除的商品编号，若编号不存在或不是该卖家所售，输出提示信息。如果商品存在，则删除该商品并提示删除成功。

(8)管理员删除买家信息、卖家信息

选择管理员身份后，选择管理员删除买家信息功能。输入要删除的买家的编号，如果买家存在，删除买家信息并提示；如果不存在，输出提示信息。

选择管理员身份后，选择管理员删除买家信息功能。输入要删除的卖家的编号，如果卖家存在，删除卖家信息并提示；如果不存在，输出提示信息。

(9)保存信息到文件

选择管理员身份后，可将买家信息、卖家信息、商品信息分别保存到txt文件中。

(10)退出系统

用户可通过输入特定指令来退出程序，在用户输入退出指令之前，程序不得自行退出。

* 1. 要求

(1)源程序编写要求

根据系统功能描述，采用模块化程序设计方法进行程序设计，要求程序结构清晰。上述各个功能模块要求分别用函数实现，在主函数中通过调用这些函数，完成系统功能的要求。代码书写要规范，有简要的注释，给出函数说明。

(2)设计报告撰写要求

设计报告内容包括题目内容和要求、总体设计、详细设计、源代码、调试过程中的问题、总结等。

总体设计：对程序的整体设计思路进行描述，画出系统的总体功能模块图，说明系统使用的主要数据结构，列表给出需要用到的函数并描述其功能。

详细设计：画出函数调用关系图，分析并描述函数的功能。

调试过程中的问题：记录程序编写和调试过程中遇到的各种问题，以及解决这些问题的途径和方法。

总结：回顾整个综合程序设计的过程，对学习到的设计方法和思路进行总结，写出个人体会。

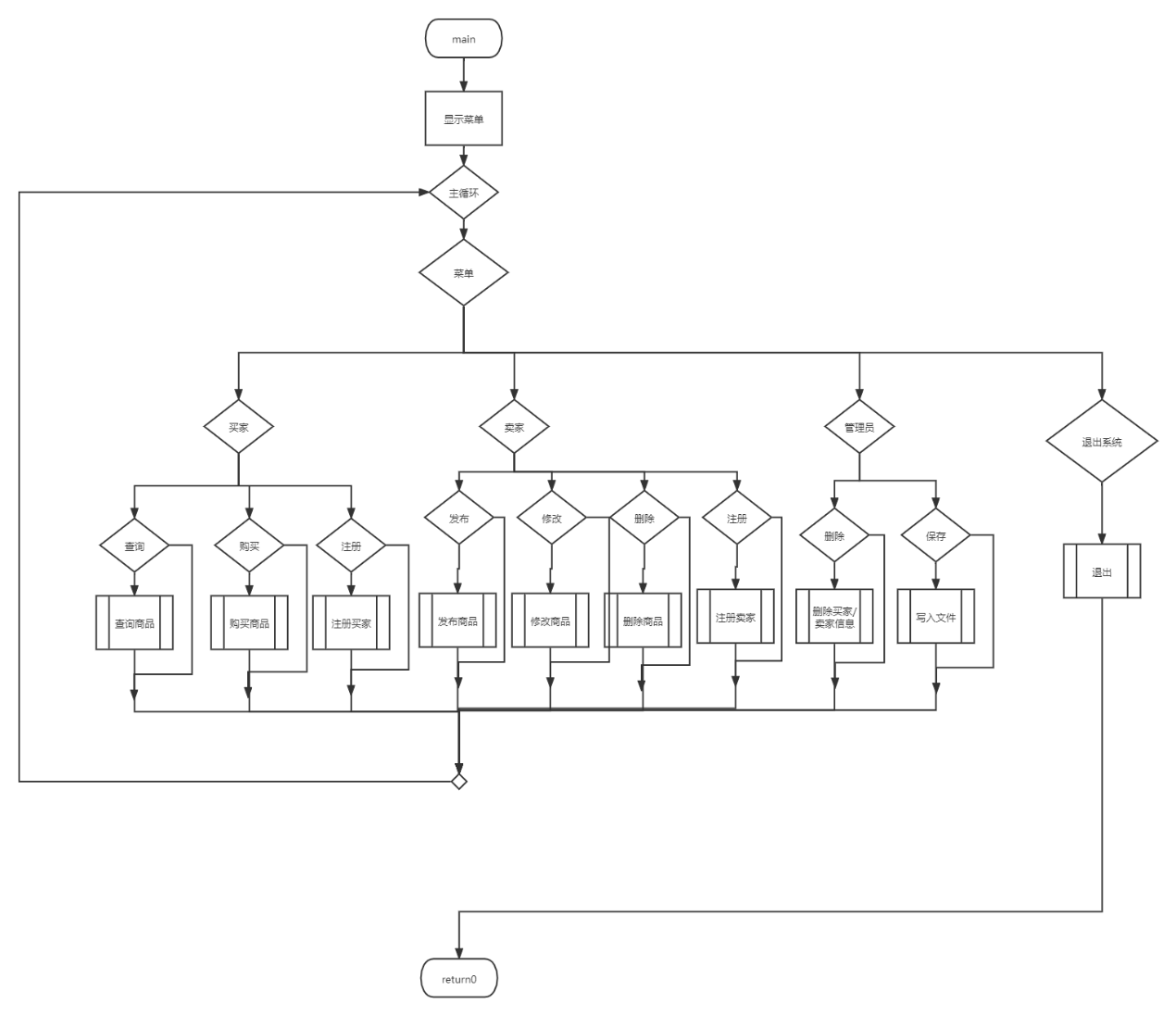
1. **问题分析**

买家、卖家、商品信息都是由多个数据项组成。在程序设计中，可以用结构体来存储，由于系统功能涉及三者的插入和删除，使用链表存储信息更有利于对于三者的管理。根据题目要求，采用单链表。结点由数据域和指针域构成。为了避免重复定义三种节点类型而产生冗余的代码，数据域可采用模板的数据类型，再通过买家、卖家、商品三种结构体的定义传入模板，实现链表的构成。指针域存储其直接后继结点的地址。

卖家是网购系统的起点，所有商品的生命周期都是从卖家发布商品开始，直至买家买完商品结束。

商品是网购系统最基础的对象，所有买家、卖家都是对商品进行操作。商品信息的查找、购买、删除、修改、插入都需要先查找，找到目标结点后再进行相应操作。对于三者的排序，可采用选择排序、冒泡排序、插入排序等，针对单链表结构的特点，网购系统可采用插入排序的方法实现按编号的排序。商品卖出时，找到目标结点后，需根据商品数量判断是否还有商品，再完成后续操作。保存信息后，所有信息被写入数据文件。

1. **总体设计**
   1. 功能模块设计

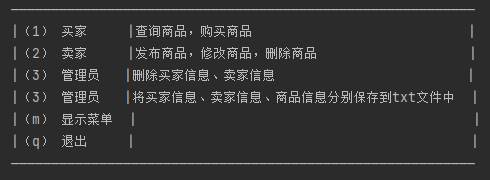


**图3.1-1 主程序流程图**

* 1. 系统界面设计

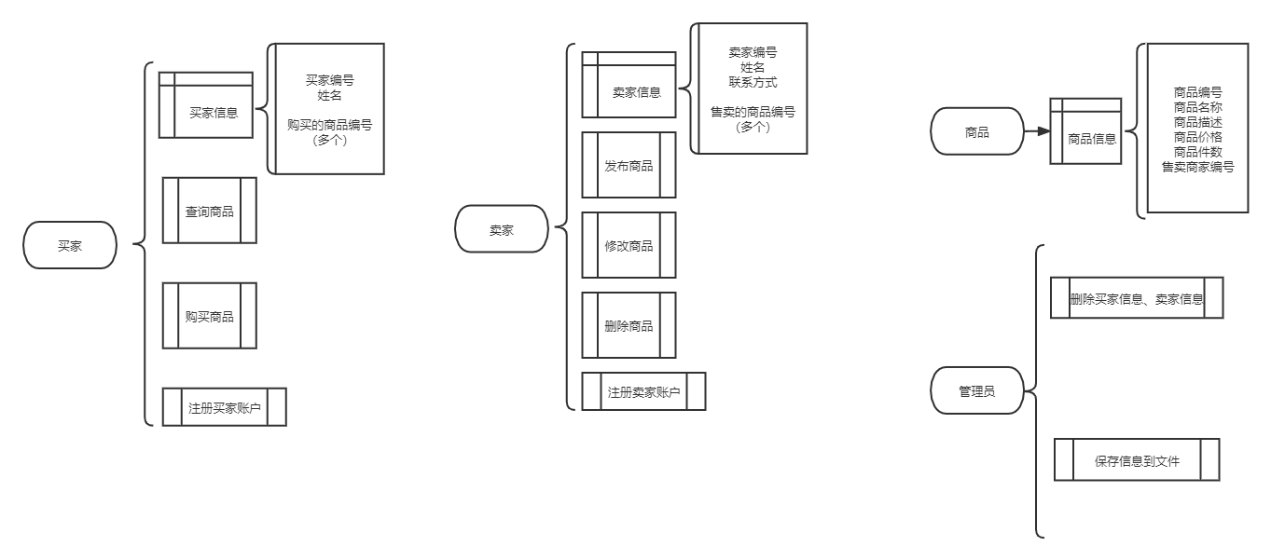
使用“|”与“—”构成UI界面。

为了便于用户使用，需要保证无论用户在那个层级都能通过指令查看当前状态的菜单。



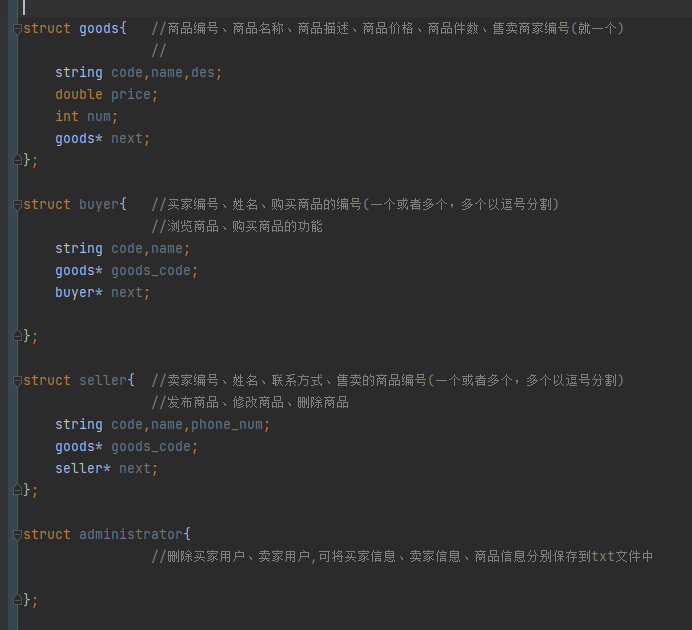
**图3.2-1 主系统界面**

* 1. 数据结构设计
     1. 链表与节点

****

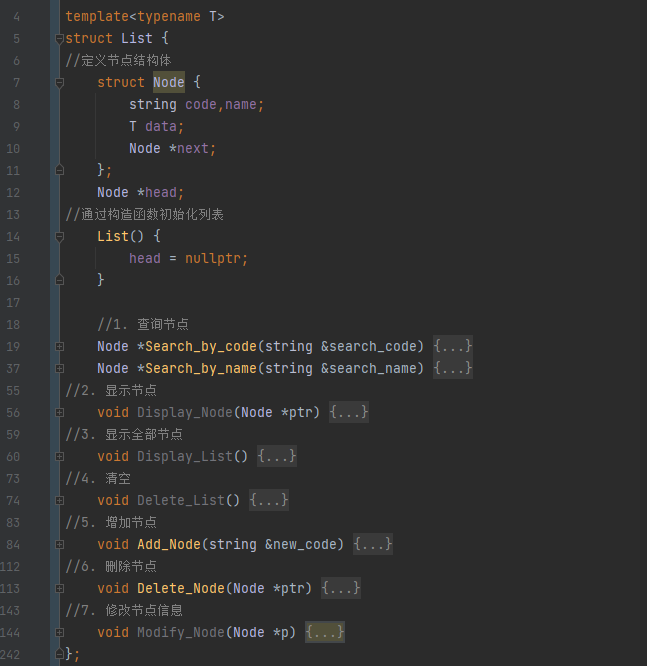
**图3.3-1 买家、卖家、商品、管理员三种结构体的导图**

在定义结构体的时候，需要对三种结构体进行的操作有很多相似的地方，但定义略显冗余。



**图3.3-2 买家、卖家、商品、管理员三种结构体的定义**

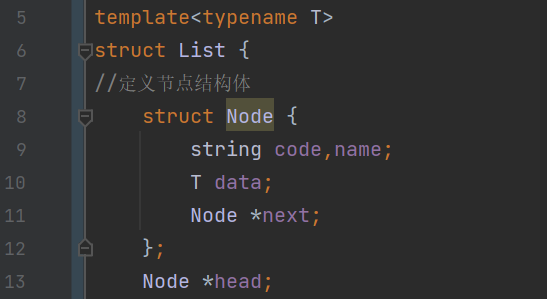
为了避免重复定义结构相似的三种链表，我希望通过将买家，卖家，商品的数据传入链表类实现三种链表的实例化。通过查阅资料**[1]**，我学习了如何使用C++模板的功能。具体实现如下图所示：



**图3.3-3 使用模板定义链表类**

由上图可见一个链表类数据包括数据域和指针域。数据域可分为共用（编号code与姓名name）和特殊数据（data）；指针域只包含一个指向下一个节点的指针（next）。

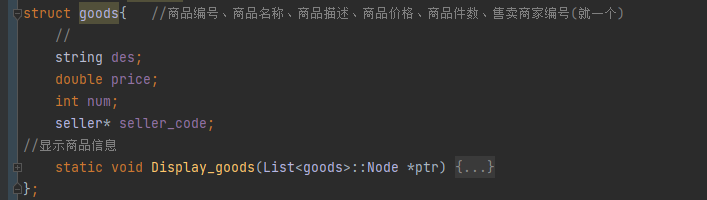
节点内数据包括使用模板，便于将商品、买家、卖家的信息以链表形式储存。



**图3.3-4 链表的结构体定义**

* + 1. 商品

商品是网购系统最基础的元素，每个商品都具备有唯一的六位数编号商品的特殊数据包括商品描述（des）、价格（price）、剩余数量（num）以及卖家的编号。其中，商品描述为字符串类型的变量；价格设置为双精度浮点，更符合商品定价的方式；卖家的编号我通过只想卖家的指针实现。此外，我将商品的操作（即显示商品信息）作为其成员函数放在该结构体内。

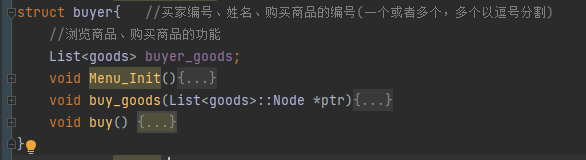


**图3.3-5 商品的结构体定义**

* + 1. 买家（buyer）

买家本身是一个链表（buyers），这样的设计便于管理员操作。

买家的特殊数据包括购买的商品编号，通过将该买家购买的所有商品放入其私有列表实现。将已购买商品放入买家的私有列表（buyer\_goods），这些商品不再公开显示，且管理员也无权限查看，这保护了买家的隐私。此外，买家的成员函数包括买家界面的初始化（Menu\_Init）、买东西（buy\_goods）和买家操作（buy）。（详见函数设计）



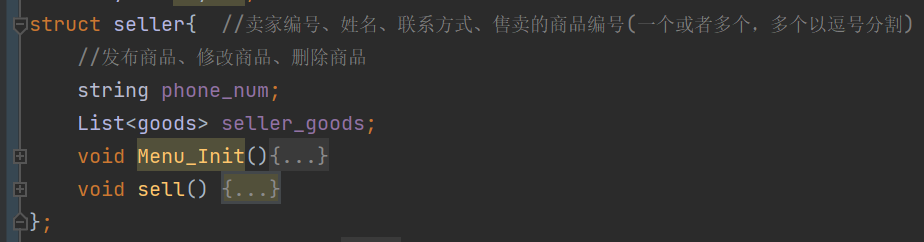
**图3.3-6 买家的结构体定义**

* + 1. 卖家（seller）

卖家本身是一个链表（sellers），这样的设计便于管理员操作。

卖家的特殊数据包括电话号码以及他所售的商品，后者通过将卖家所售商品放入其私有列表（seller\_goods）实现。同时，卖家发布的商品会在公开市场中显示，这代表不同卖家如果卖相同商品将拥有唯一的商品编号，避免买家无法挑选商家的问题。

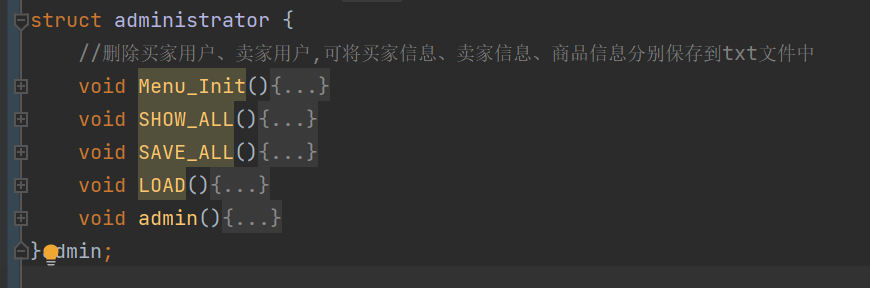
此外，买家的成员函数包括卖家界面的初始化（Menu\_Init）和卖家操作（sell）。（详见函数设计）



**图3.3-7 卖家的结构体定义**

* + 1. 管理员（administrator）

管理员没有数据，仅包括操作。因此，其结构体内只有管理员操作，其中包括管理员界面的初始化（Menu\_Init）、显示全部成员信息（SHOW\_ALL）、保存所有信息到文件（SAVE\_ALL）、从文件中加载所有信息（LOAD）和管理员操作（admin）。



**图3.3-8 管理员的结构体定义**

* 1. 函数设计

1. 链表与节点（List and Node）

通过链表类的构造函数初始化链表第一个节点的头指针：

List() {

head = nullptr;

}

查询节点有通过编号查询以及通过名称查询两种方式。其中编号查询多用于编辑商品的内部操作，而名称查询则是面向买家的用户接口。

//1. 查询节点  
 Node \*Search\_by\_code(string &search\_code) {  
 Node \*ptr = head;  
 if (!ptr) //链表为空，直接返回

{return nullptr; } else {  
//判断是否为查找节点  
 while (ptr) {  
 if (search\_code == ptr->code) {return ptr; }  
 ptr = ptr->next;  
 }  
 return nullptr; }}  
 Node \*Search\_by\_name(string &search\_name) {  
 Node \*ptr = head;  
 if (!ptr) //链表为空，直接返回

{return nullptr; }

else {  
//判断是否为查找节点  
 while (ptr) {  
 if (search\_name == ptr->name) {return ptr; }  
 ptr = ptr->next; }  
 return nullptr; }

}

1. 商品（goods）

商品是网购系统最基础的元素，但其本身并没有任何功能上的需求。但是为了后续某些操作的方便，其中包含一个显示商品详细信息的函数。

1. 买家（buyer）

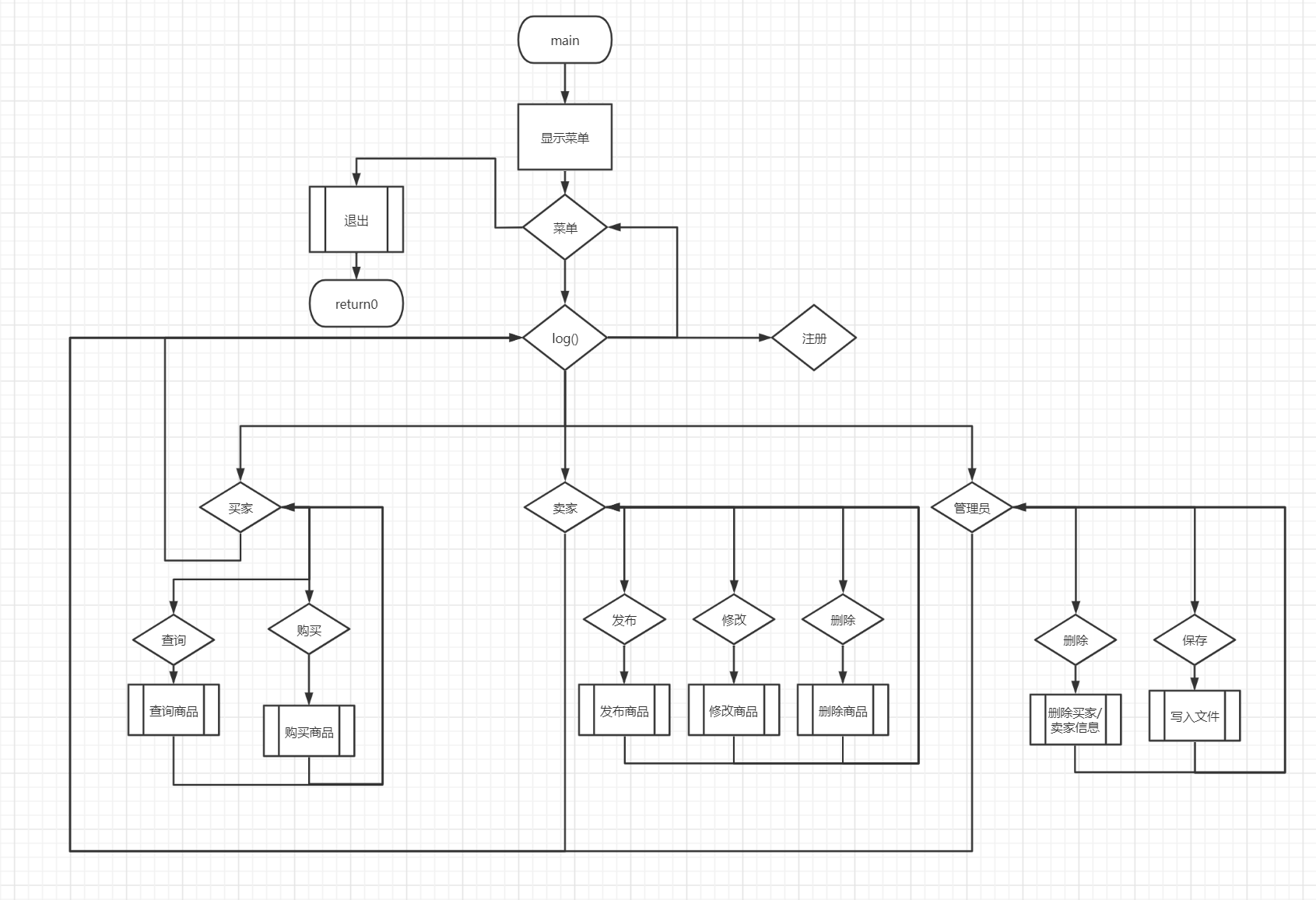
买家对于商品的操作是作为顾客的角度。我们应当考虑，所有产品的信息中：商品的名称是顾客接触的到的，而其他信息是顾客接触不到的，但根据需求，我们应当向他们展示，但具体对商品的操作应当由系统来完成，而并非买家决定。因此，买家的操作限制很多。

1. 在查询过程中，我们应当考虑买家想要知道的信息有哪些,以及不同卖家可能发布同名的商品。
2. 在购买过程中，我们应当考虑商品是否处于可以被交易的状态（主要考虑受商品数量、卖家状态限制）。
3. 卖家（seller）

卖家是网购系统的起点，所有商品的生命周期都是从卖家发布商品开始，直至买家买完商品结束。

1. 卖家发布商品，需要考虑市场上有没有相同的编号，所有商品有且仅有一个编号
2. 卖家修改商品时，可分别修改商品数据，其他商品数据默认保持不变。
3. 卖家删除商品需保证手上商品与市场上商品同步删除。
4. 管理员（administrator）
5. 管理员保存文件时，需要考虑保存信息之后的读取。必须将文件保存与载入的数据接口相匹配才能保证整个系统运行的稳定性。
6. **详细设计**
   1. 各个函数的调用关系图

如图所示：



**图4.1-1 主函数与子函数调用关系流程图**

* 1. 各个函数的功能描述

4.2.1初始化系统界面

用最基础的cout以及“-”“|”构成系统界面。

void Menu\_Init() {  
 cout << "——————————————————————————————————————————————————————————" << endl  
 << "|（1） 买家 |查询商品，购买商品 |" << endl  
 << "|（2） 卖家 |发布商品，修改商品，删除商品 |" << endl  
 << "|（3） 管理员 |删除买家信息、卖家信息 |" << endl  
 << "| |将买家信息、卖家信息、商品信息分别保存到txt文件中 |" << endl  
 << "|（m） 显示菜单 | |" << endl  
 << "|（q） 退出 | |" << endl  
 << "——————————————————————————————————————————————————————————" << endl;  
}

4.2.2 具体功能的实现

1. 商品（goods）

商品是网购系统最基础的元素，但其本身并没有任何功能上的需求。但是为了后续某些操作的方便，我写了一个显示商品详细信息的函数：

static void Display\_goods(List<goods>::Node \*ptr) {  
 if(!ptr){  
 cout << "无商品" << endl;  
 return;  
 }  
 cout << ptr->code << endl;  
 cout << ptr->name << endl;  
 cout << ptr->data.des << endl;  
 cout << ptr->data.price << endl;  
 cout << ptr->data.num << endl;  
 cout << ptr->data.seller\_code << endl;

（为了便于其他类调用，我将该函数定义为静态函数）

1. 买家

买家须实现的功能包括查询商品和购买商品。但除此之外我们还需要为买家提供一个独立的用户界面，代码如下：

void Menu\_Init(){  
 cout << "————————————————————————————————" << endl  
 << "|（1） 查询商品 |" << endl  
 << "|（2） 购买商品 |" << endl  
 << "|（m） 显示菜单 |" << endl  
 << "|（q） 退出 |" << endl  
 << "————————————————————————————————" << endl;  
}

查询商品时候需要考虑如果有重名的商品，则需要将信息全部显示，代码如下：

void search\_goods(string search\_name){  
 List<goods>::Node \*ptr = sellers\_goods.head;  
 while (ptr) {  
 if (search\_name == ptr->name) {  
 ptr->data.Display\_goods(ptr);  
 }  
 ptr = ptr->next;  
 }  
 return;  
}

购买商品过程中需要判断商品是否处于可以被购买的状态以及购买后需从卖家的手上（以及公开市场中）修改该商品的数量，代码如下：

void buy\_goods(List<goods>::Node \*ptr,int num){  
 if(!ptr){  
 cout << "无商品" << endl;  
 return;  
 }  
 if(!sellers.Search\_by\_code(sellers\_goods.Search\_by\_code(ptr->code)->data.seller\_code)){  
 cout << "卖家已销户" << endl;  
 return;  
 }  
 if(sellers\_goods.Search\_by\_code(ptr->code)->data.num<num){  
 cout << "商品货量不足" << " 商品货量仅剩：" <<sellers\_goods.Search\_by\_code(ptr->code)->data.num<< endl;  
 return;  
 }  
 if(!buyer\_goods.Search\_by\_code(ptr->code))  
 {  
 buyer\_goods.Add\_Node(ptr->code);  
 buyer\_goods.Search\_by\_code(ptr->code)->name=ptr->name;  
 buyer\_goods.Search\_by\_code(ptr->code)->data.des=ptr->data.des;  
 buyer\_goods.Search\_by\_code(ptr->code)->data.price=ptr->data.price;  
 buyer\_goods.Search\_by\_code(ptr->code)->data.num=0;  
 buyer\_goods.Search\_by\_code(ptr->code)->data.seller\_code=ptr->data.seller\_code;  
 }  
 buyer\_goods.Search\_by\_code(ptr->code)->data.num+=num;  
 sellers\_goods.Search\_by\_code(ptr->code)->data.num-=num;  
 refresh(ptr->code);  
}

1. 卖家

卖家代码同样包括独立的初始化界面。

卖家处理的关键在于其发布商品、修改商品后要将市场上的对应商品（sellers\_goods）与之同步。以发布商品为例：

if(str.length()==6) {  
 if (!sellers\_goods.Search\_by\_code(str)) {  
 cout << "您要发布的商品名字是：" << endl;  
 cin >> name;  
 sellers\_goods.Add\_Node(str);  
 sellers\_goods.Search\_by\_code(str)->data.seller\_code = seller\_code;  
 sellers\_goods.Search\_by\_code(str)->name = name;  
 seller\_goods.Add\_Node(str);  
 seller\_goods.Search\_by\_code(str)->data.seller\_code = seller\_code;  
 seller\_goods.Search\_by\_code(str)->name = name;  
 sellers\_goods.Search\_by\_code(str)->data.goods\_init();  
 seller\_goods.Search\_by\_code(str)->data.des = sellers\_goods.Search\_by\_code(  
 str)->data.des;  
 seller\_goods.Search\_by\_code(str)->data.price = sellers\_goods.Search\_by\_code(  
 str)->data.price;  
 seller\_goods.Search\_by\_code(str)->data.num = sellers\_goods.Search\_by\_code(  
 str)->data.num;  
 } else {  
 cout << "商品已存在" << endl;  
 }  
}

1. 管理员

管理员同样包括独立的初始化界面。

为了便于对系统进行维护，管理员具有查看所有信息的权限：

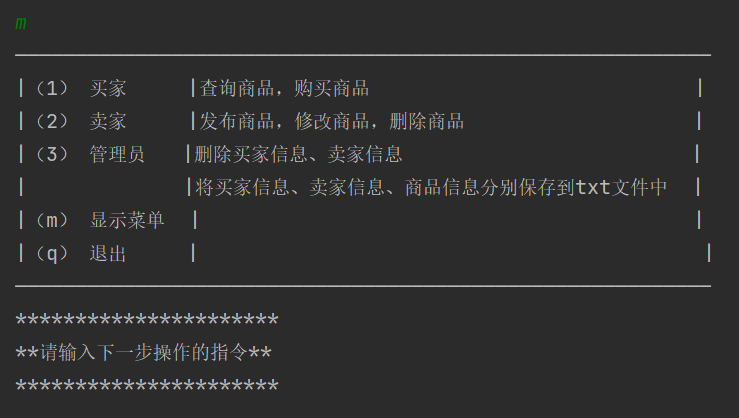
void SHOW\_ALL(){  
 sellers\_goods.Display\_List();  
 sellers.Display\_List();  
 buyers.Display\_List();  
}

加载文件数据与保存数据到文件都是管理员的操作。保存文件时，需要考虑保存信息的读取。必须将文件保存与载入的数据接口相匹配才能保证整个系统运行的稳定性（这里仅以保存文件的代码为例）：

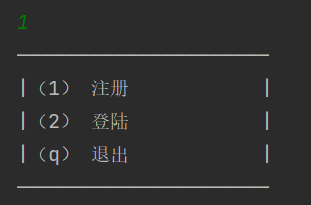
void SAVE\_ALL(){  
 ofstream ofs\_buyer,ofs\_seller,ofs\_goods;  
 ofs\_goods.open("goods.txt",ios::out);  
 ofs\_buyer.open("buyer.txt",ios::out);  
 ofs\_seller.open("seller.txt",ios::out);  
//保存商品  
 List<goods>::Node \*goods\_h=sellers\_goods.head;  
 while(goods\_h){  
 ofs\_goods<<goods\_h->code<<' '<<goods\_h->name<<' '<<goods\_h->data.des<<' '<<goods\_h->data.price<<' '<<goods\_h->data.num<<' '<<goods\_h->data.seller\_code<<endl;  
 goods\_h=goods\_h->next;  
 }  
//保存买家  
 List<buyer>::Node \*buyer\_h=buyers.head;  
 while(buyer\_h){  
 ofs\_buyer<<buyer\_h->code<<' '<<buyer\_h->name;  
 if(buyer\_h->data.buyer\_goods.head){  
 List<goods>::Node \*buyer\_goods\_h=buyer\_h->data.buyer\_goods.head;  
 while(buyer\_goods\_h) {  
 ofs\_buyer <<" ,";  
 ofs\_buyer << ' ' << buyer\_goods\_h->code << ' ' << buyer\_goods\_h->name << ' ' << buyer\_goods\_h->data.des<< ' '  
 << buyer\_goods\_h->data.price << ' ' << buyer\_goods\_h->data.num << ' '<< buyer\_goods\_h->data.seller\_code;  
 buyer\_goods\_h = buyer\_goods\_h->next;  
 }  
 }  
 ofs\_buyer<<endl;  
 buyer\_h=buyer\_h->next;  
 }  
//保存卖家  
 List<seller>::Node \*seller\_h=sellers.head;  
 while(seller\_h){  
 ofs\_seller << seller\_h->code << ' ' << seller\_h->name << ' ' << seller\_h->data.phone\_num;  
 if(seller\_h->data.seller\_goods.head){  
 List<goods>::Node \*seller\_goods\_h = seller\_h->data.seller\_goods.head;  
 while (seller\_goods\_h) {  
 ofs\_seller <<" ,";  
 ofs\_seller << ' '<< seller\_goods\_h->code << ' ' << seller\_goods\_h->name << ' ' << seller\_goods\_h->data.des  
 << ' '<< seller\_goods\_h->data.price << ' ' << seller\_goods\_h->data.num << ' '<< seller\_goods\_h->data.seller\_code;  
 seller\_goods\_h = seller\_goods\_h->next;  
 }  
 }  
 ofs\_seller<<endl;  
 seller\_h = seller\_h->next;  
  
 }  
 ofs\_goods.close();  
 ofs\_buyer.close();  
 ofs\_seller.close();  
 }

1. **功能测试**
2. 初始化菜单

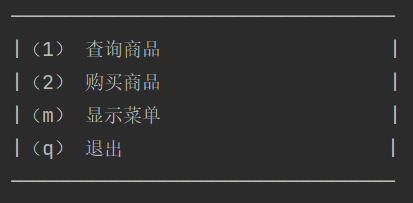
根据代码设计，用户可以在任何时候通过“m”键查看对应界面下的菜单。



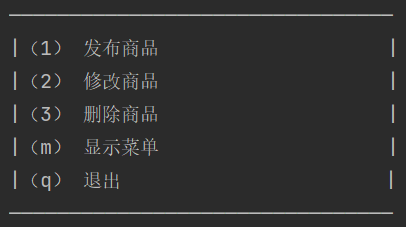
**5.1-1 系统主菜单界面**



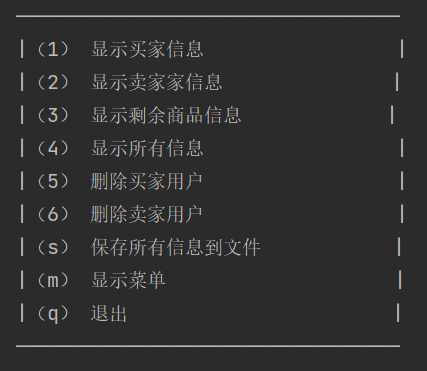
**5.1-2 用户注册/登录界面**



**5.1-3 买家界面**



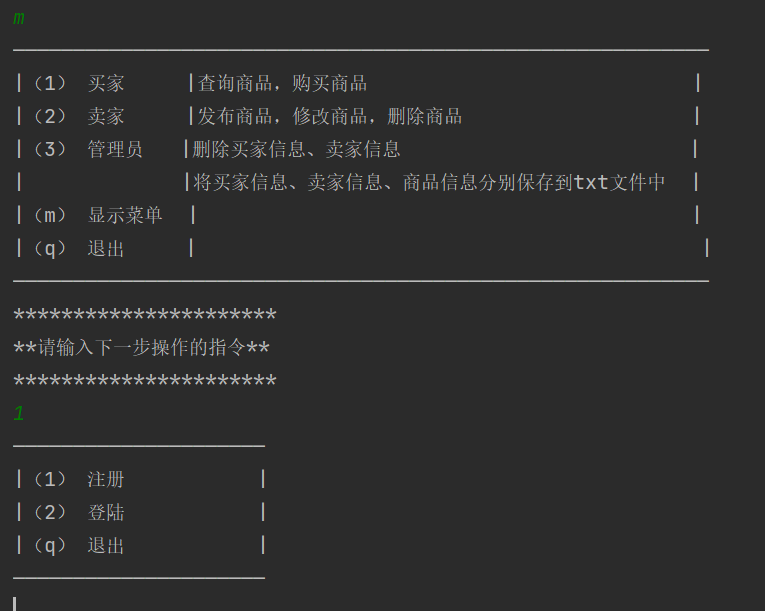
**5.1-4 卖家界面**



**5.1-4 管理员界面**

1. 选择用户身份权限

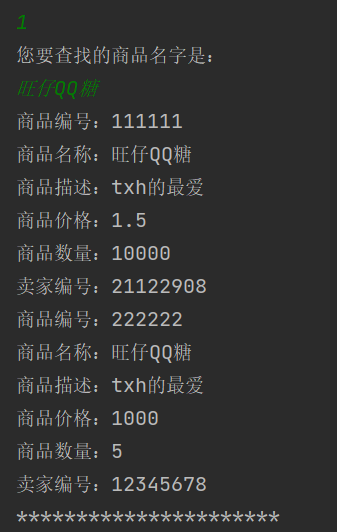
可以通过菜单进行用户身份选择，但实际用户身份是通过注册/登录的账号来决定所获得的权限。买家账号为七位数，卖家账号为八位数。



**5.2-1 伪用户身份选择界面**

1. 买家查询商品

查询商品主要考虑了不同卖家的同名商品需要全部显示。



**图5.3-1 买家查询存在同名商品时，全部显示**

1. 买家购买商品

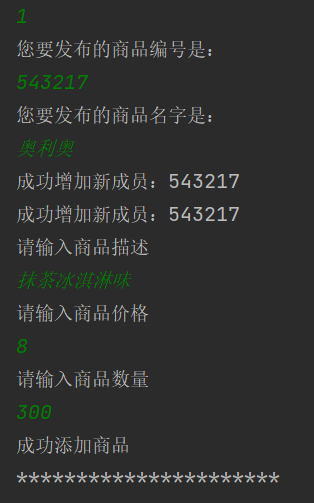
买家购买商品就是将商品加入自己的私有链表（buyer\_goods），成功添加后会有文字提示。



**图5.4-1 买家购买商品**

1. 卖家发布商品

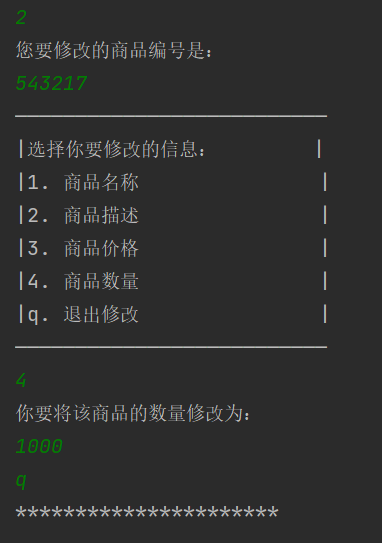
卖家发布商品时，需根据提示初始化商品信息，以保证商品信息完整。



**图5.5-1 卖家发布商品**

1. 卖家修改商品

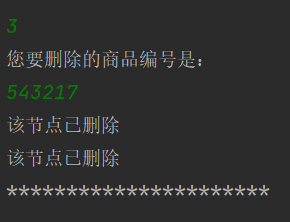
卖家可以根据自己需求选择并修改商品信息。其他信息将保持原数据不做更改最终按q可保存并退出修改。



**图5.6-1 卖家修改商品**

1. 卖家删除商品

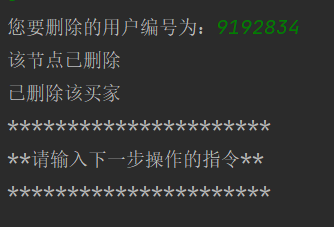
卖家删除商品时必须根据商品唯一的编号，以防止误删。



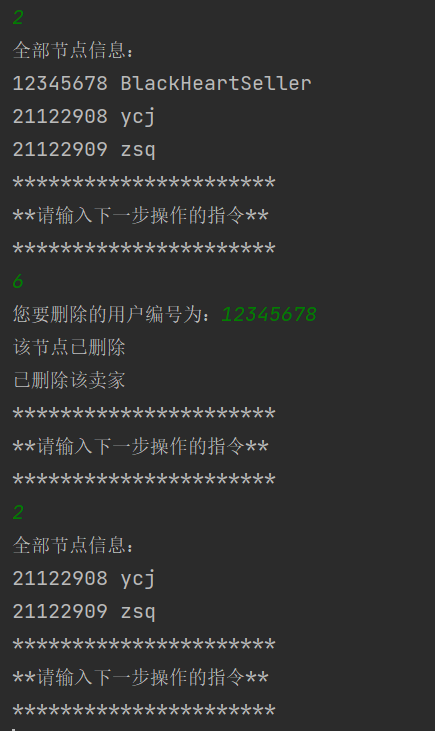
**图5.7-1 卖家删除商品**

1. 管理员删除买家信息、卖家信息

管理员删除商买/卖家信息时必须根据买/卖家唯一的编号，以防止误删。

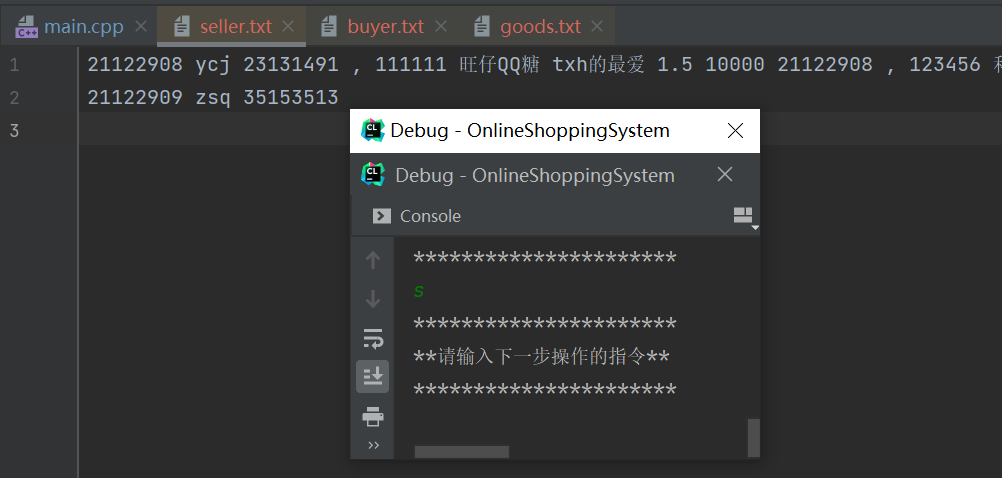


**图5.8-1 管理员删除买家信息**

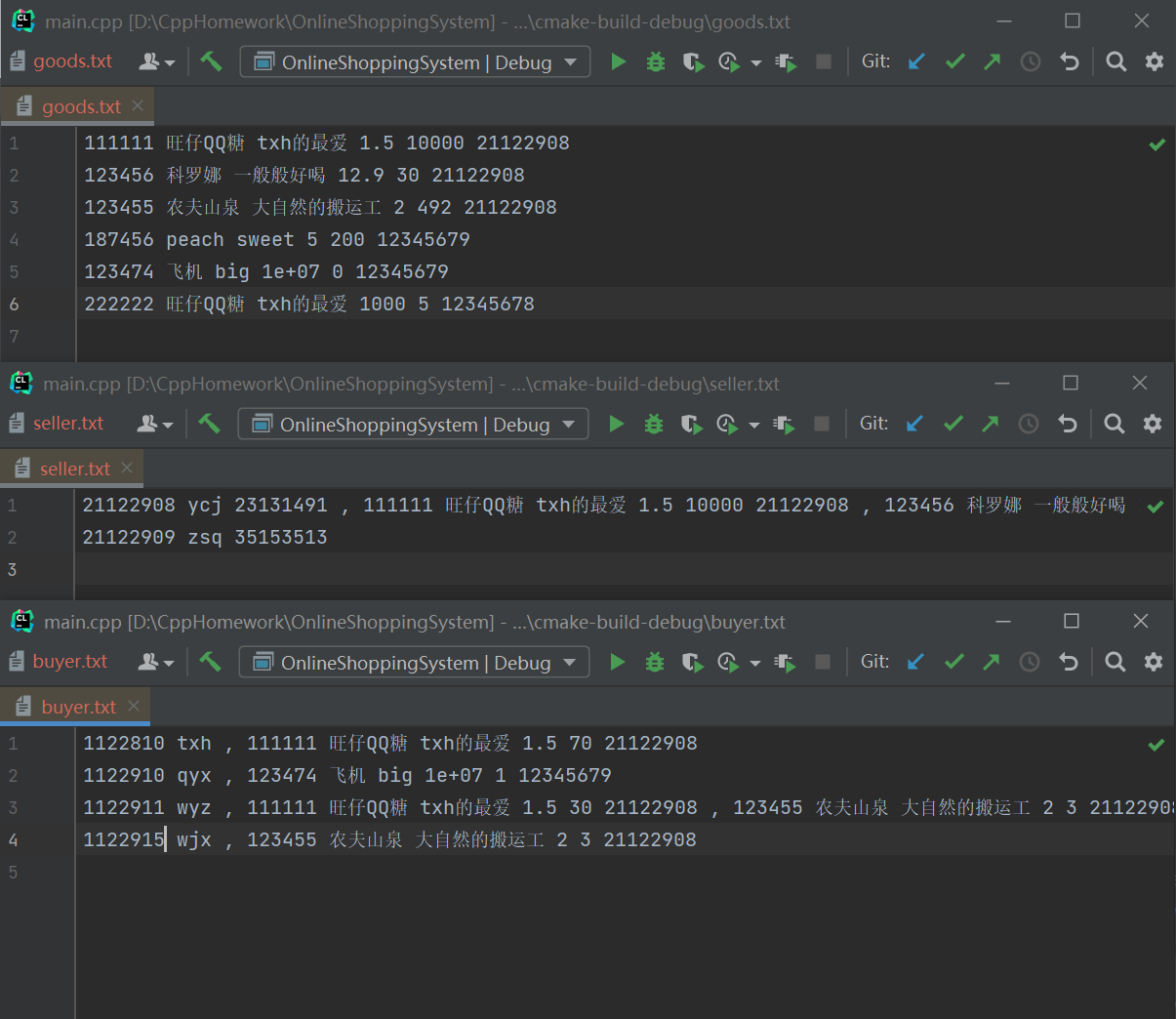


**图5.8-2 管理员删除卖家信息**

1. 保存信息到文件



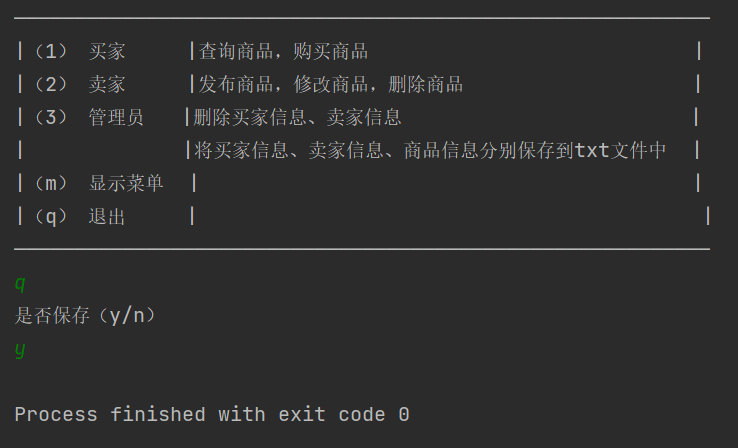
**图5.9-1 管理员保存所有信息**



**图5.9-2保存文件格式**

1. 退出系统

主菜单按“q”即可退出程序

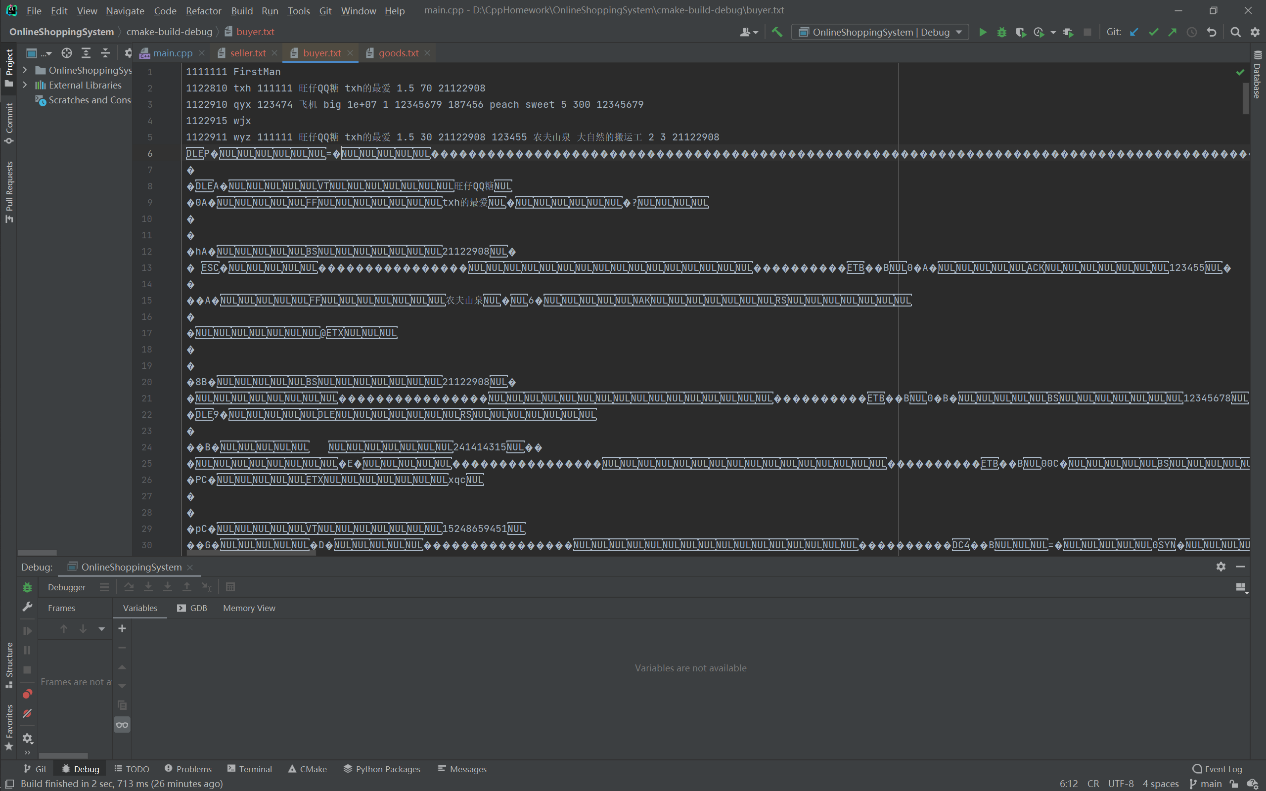


**图5.10-1退出系统**

1. 总 结

在设计数据结构时，为了避免重复定义结构相似的三种链表，我希望通过将买家，卖家，商品的数据传入链表类实现三种链表的实例化。在此过程中，我通过查阅资料，学习了如何使用C++类与对象的概念以及模板的功能。

此外在调试过程中，我认识到对于链表操作最容易犯的错误是对节点指针的操作。



**图6-1 节点指针指向链表外数据导致文件读写出错**

1. 参考资料

[1] <https://www.runoob.com/cplusplus/cpp-templates.html>