





|  |  |  |  |
| --- | --- | --- | --- |
| 学号 |  | 姓名 |  |
| 班级 |  | 任课教师 |  |
| 实验名称 | 网络购物系统的设计 | | |
| 实验学期 | 2023-2024学年第1学期 | | |
| 实验日期 | 2023年12月23日 | 实验地点 | G楼320 |
| 报告成绩 |  | 评分教师 |  |

西安电子科技大学计算机科学与技术学院

1. **实验环境**

Visual studio 2022

处理器 12th Gen Intel(R) Core(TM) i5-1240P 1.70 GHz

机带 RAM 16.0 GB (15.6 GB 可用)

设备 ID 5937A78D-ADDC-4537-A9AA-31E44F288BCD

系统类型 64 位操作系统, 基于 x64 的处理器

1. **实验内容**

**1. 实现登录功能并管理顾客的信息**

（1）管理员必须登录系统之后才能操作后台数据；管理员的账户和密码通过提前预设的方式指定。

（2）顾客未登录系统时，可以查询商品信息。若要购买商品，必须登录系统。顾客在首次登录系统时，需要先进行注册，注册成功后设置账户和密码。

（3）保存顾客注册的信息和账户信息；

（4）顾客可以修改自己的密码。

**2.管理员的功能：增删改**

（1）能够根据管理员的输入，添加新的商品信息；

（2） 能够删除指定商品的信息；

（3） 能够修改指定商品的信息；

**3. 顾客的功能**

（1）能够根据用户给出的商品名称查询商品信息，并显示查询结果；

（2）能够根据用户给出的关键字，给用户推荐最接近的商品信息。

（3）顾客能够选择购物车中的商品进行结算，计算并显示结算商品的总价；

（4）从购物车中删除已经结算的商品。

**4. 购物车的功能同时具有持久化与恢复的特征**

（1）为每个顾客创建购物车；

（2）根据顾客的输入，向购物车中添加商品；

（3）顾客能够查询，删除，修改购物车中的商品信息；

（4）能够保存顾客的购物历史记录，包括购买的商品和购买时间。根据顾客选定的时间，按照所购买商品的种类展示购买的金额。

（5）能够计算购物车中所有商品的总价，并将总价显示给用户；

**5. 优惠与折扣**

（1）给顾客随机派送优惠券（优惠券面值也可以随机），顾客结算商品时可以输入优惠券码，对结算总价进行折扣计算。

（2） 活动折扣。特定优惠活动的折扣，结算时按折扣规则进行折扣计算。

1. **实验步骤**

**框架构建：**根据主要的需求划分，我定义了四个核心的类，用户，产品，购物商场以及购物车。管理员与顾客分别继承父类人的信息，并添置自己需要的属性与功能。购物商场围绕产品类展开，购物车以购物商场为基础添置商品。

**需求实现**：

1.对于登录功能，为了实现顾客信息的持久化，我将顾客的用户名，密码，姓名，住址，这些具有用户特征的信息储存至文件中，用add的形式进行追加。当用户登录成功后，将信息与currentuser进行绑定，以便后续操作。对于管理员的登录，采用预先指定的方式，当成功登录后，currentuser的身份即为管理员，不能越界进行顾客的操作。

2.对于管理员的增修改功能，本质上对文件中有关物品的信息进行改动。如果需要添加新的商品，则追加。对于删除与修改的操作，采用遍历文件内容，将所需信息添置到新的文件中，再重新命名为源文件名。后续有关文件的操作基本一致。

3.对于购物商场功能，实则为打印商品信息的界面。将储存在文件中的产品信息添置到vector<product>product中，方便遍历打印，以及转移到储存购物车商品的vector<product>shproduct中，为后续购物车功能的实现打下基础。

4.对于购物车功能，为了实现购物车顾客差异化，我将购物车文件以用户名进行命名。为了实现购物历史查询以及数据分析的功能，我又建立了一个向量vector<PurchaseHistory>，包含产品以及购买时间信息。以北京时间作为购买时间，将所购买的产品加入向量，并写入文件中，以用户名加上pur的后缀命名实现永久保存。

5.对于顾客的功能，在购物车文件的基础上，实现永久化以及增删改的基础功能。结算商品需要确认支付密码与currentuser的密码是否匹配。对于查询商品功能，可以通过直接搜索和通过find()函数进行关键词查找的方式进行展示。

6.对于优惠与折扣的功能，我利用数组coupons[couponCode] = discount;，将输入优惠券内容与对应折扣值对应起来。当顾客输入，I Love XD时便会通过在0-0.2的随机数对商品进行优惠折扣。



**问题与解决方案：**

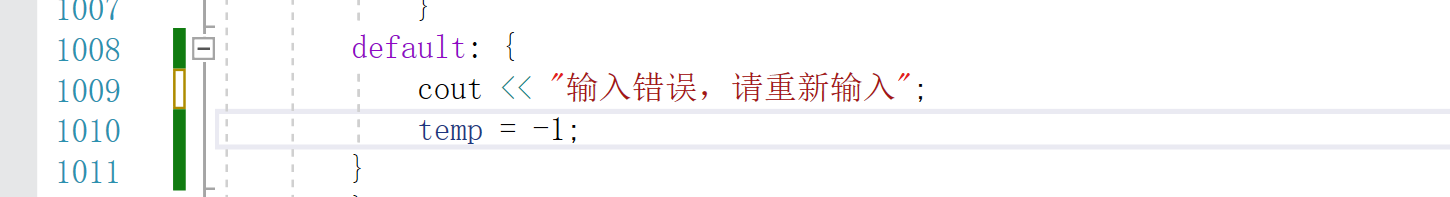
1.文件信息的提取与储存。文件每一行往往携带多个信息，或用空格分割，或用“，”分割。利用getline()提取每一行内容，如果以空格分割，用istringstream iss()遍历提取每一项内容，储存到向量中。如果以逗号分割，则以逗号定位下一项内容的位置。例如，price = stod(line.substr(pos + 1));表示price位于第一个逗号的后面的内容，pos = line.find(",", pos + 1);表示第二个逗号，num = stoi(line.substr(pos + 1));则表示第二个逗号后面的内容。

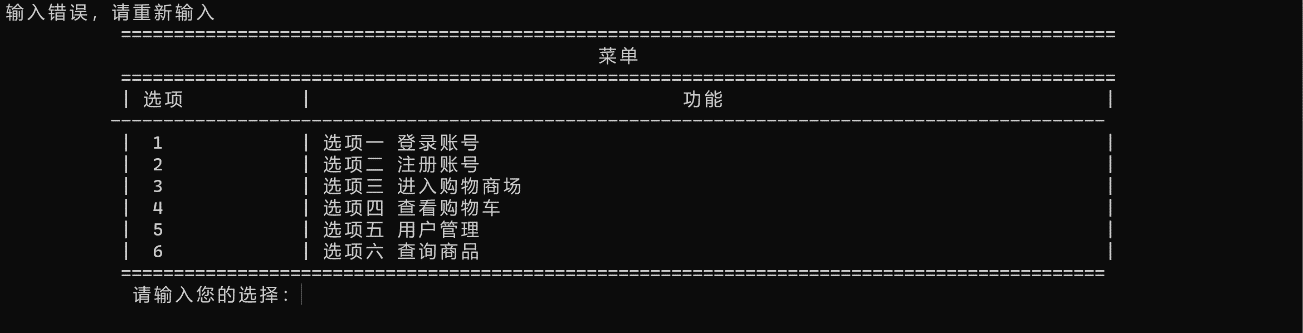
2.文件信息各项类型不一致。当每一行信息既有string，又有double，又有int时。用stoi()将string类型转化为int型。

3.文件内容与向量内容的同步。每次对于用户信息，商品信息的调整，都在文件中进行删改，而操作方法的对象是向量。需要额外写一个函数，使文件向vector传递内容，实现恢复购物车的功能。

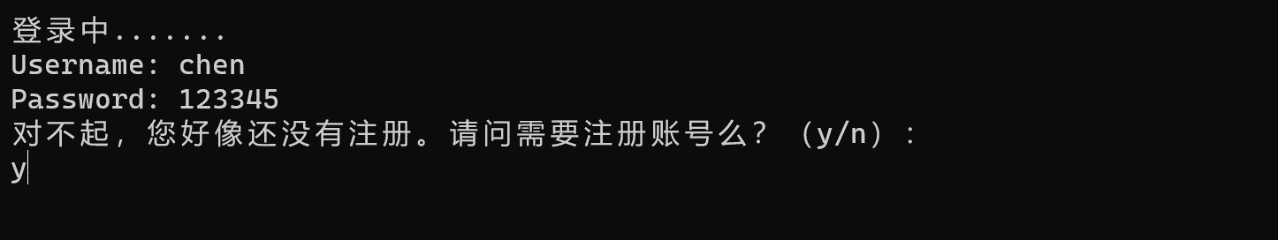
4.在用户进入购物车等与用户账户相关的内容时，如果用户未登录甚至未注册，需要页面跳转。用switch，改变当前temp值，进入对应菜单内容。

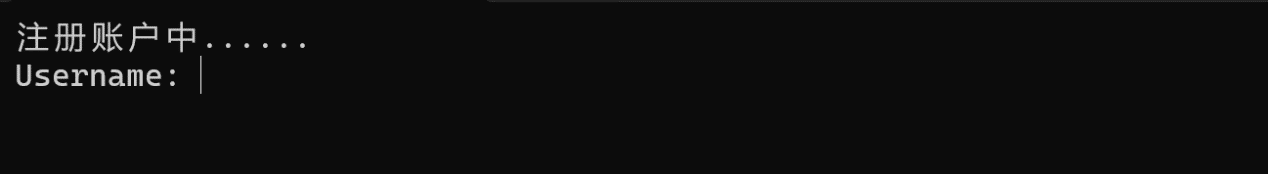
1. **实验结果**

* 菜单界面误输



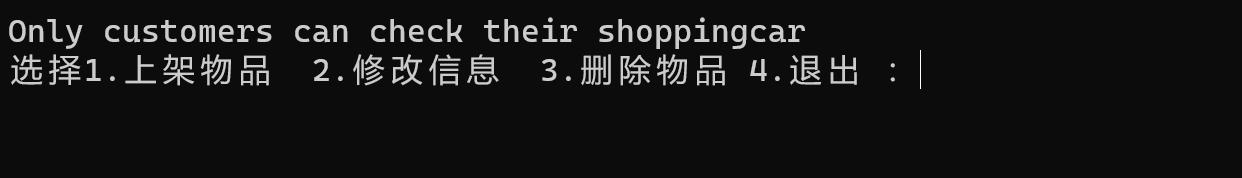
* 用户未注册过账号



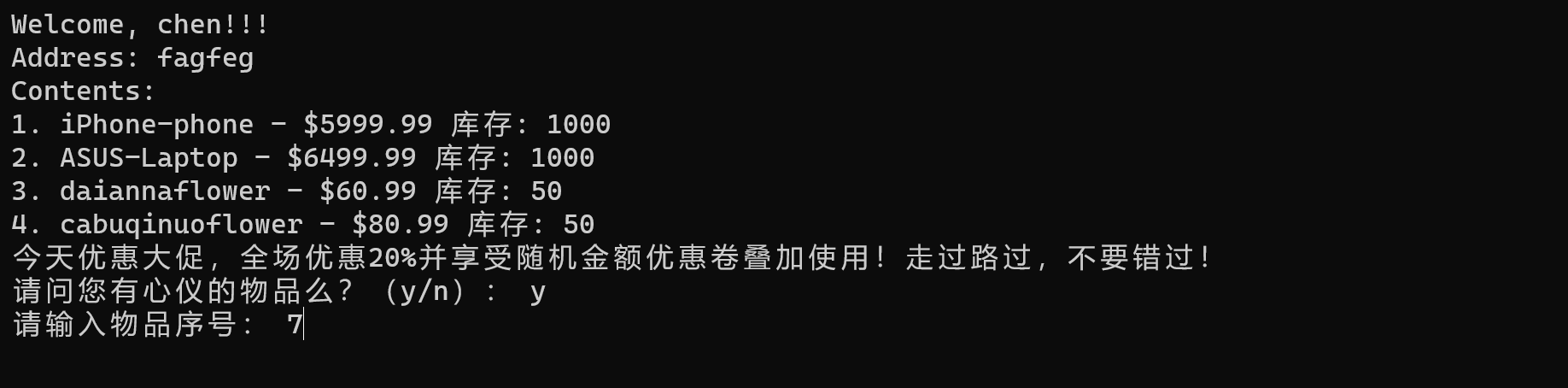


* 管理员错误操作





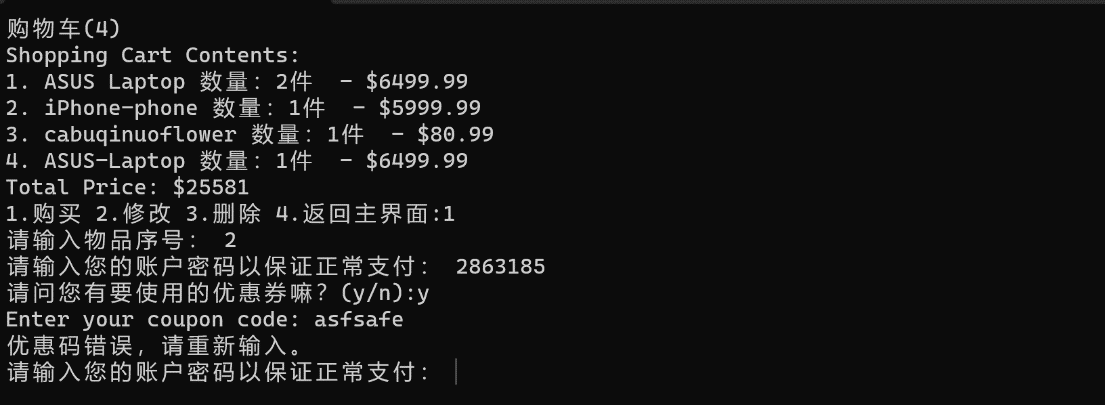
* 加入购物车序号越界



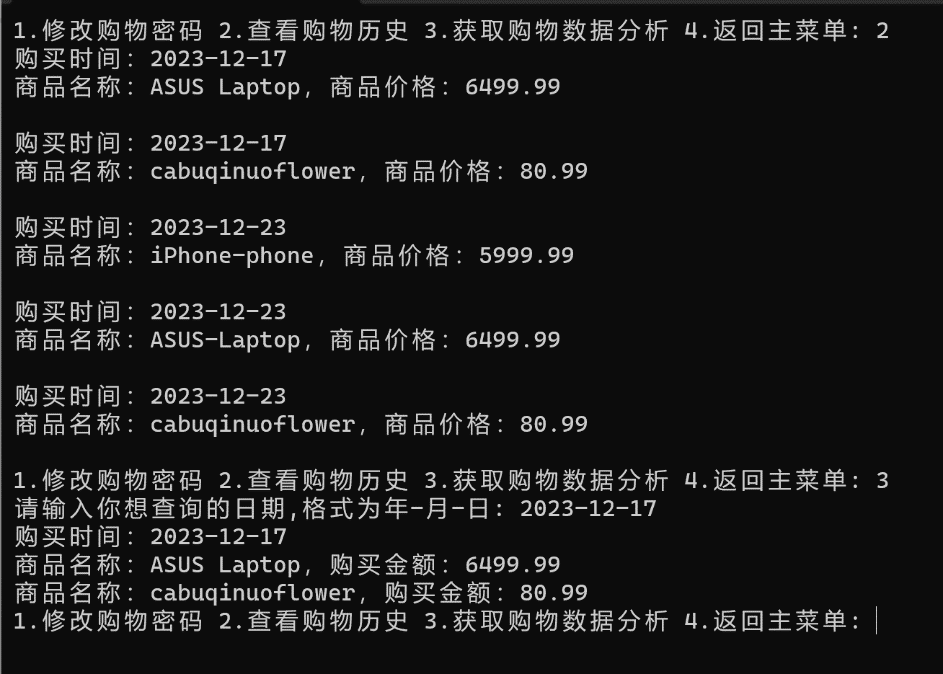


* 购买时密码错误或者优惠劵码错误





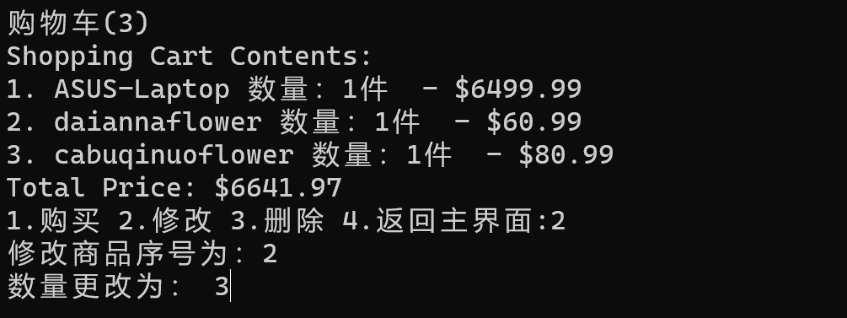
* 展示购物历史以及数据分析

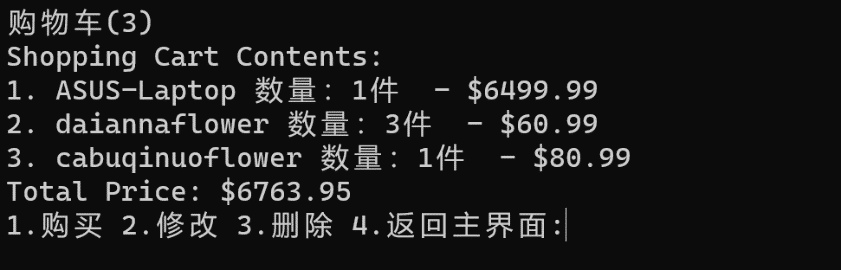


* 购物车页面展示

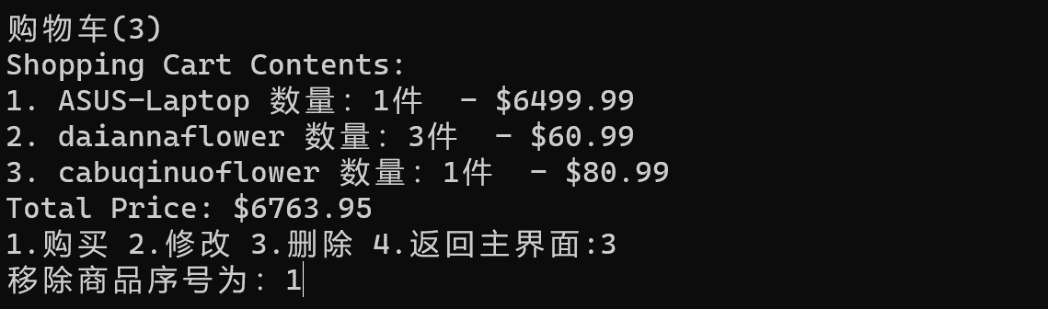


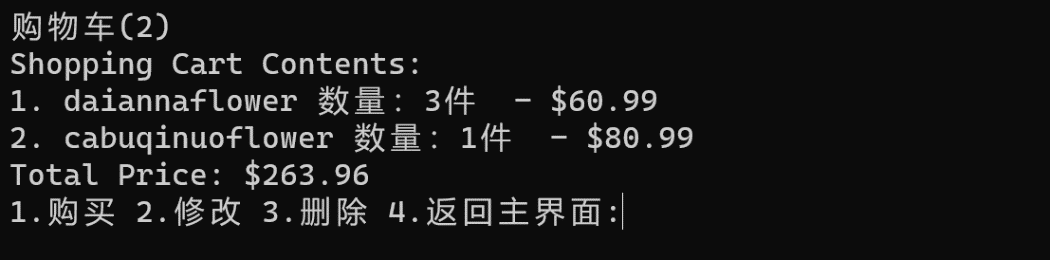
修改功能



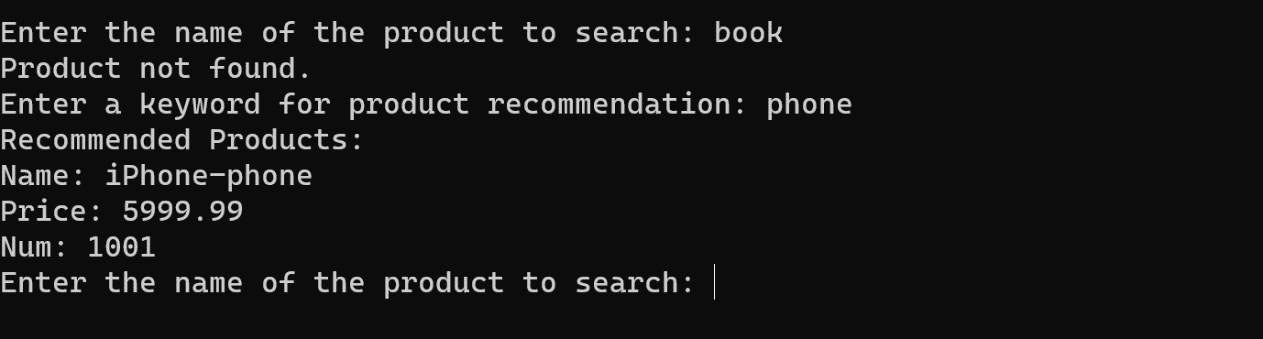


删除功能





* 查询商品



1. **实验总结**

通过这次实现网上购物系统面对对象程序设计范型，我较为深刻的理解了模块化与封装对代码可维护性和可扩展性的重要作用。通过面向对象程序设计，我们可以将问题和解决方案转化为对象和对象之间的交互。这种抽象和建模能力是培养分析和解决问题的能力的重要一步。而在具体的实现需求扩展中，如何在现有基础上进行扩展更体现了设计分析的重要性。与文件相关的功能实现，一定程度上很好的填补了我在此方面能力的空缺。而对边界点的一次次探索，也完善了我在设计代码前对各种情况应有的考虑。总而言之，此次实验培养了我良好的设计和分析能力，通过不断练习和实践，一定会提高代码能力，编写出高效、可维护和可扩展的代码。

**附录 源程序清单（要求注释不少于源代码行数的三分之一）**

#define \_CRT\_SECURE\_NO\_WARNINGS 1

#include <iostream>

#include<iomanip>

#include <vector>

#include <any>

#include<map>

#include <cstdlib>

#include <Windows.h>

#include<string>

#include<fstream>

#include<sstream>

#include <random>

#include <unordered\_map>

using namespace std;

//user类可被Customer，Admin继承，作为他们的父类

class User {

protected:

string username;

string password;

public:

User(string username, string password) {

this->username = username;

this->password = password;

}

string getUsername() {

return username;

}

string getPassword() {

return password;

}

void setPassword(string newPassword) {

password = newPassword;

}

};

//顾客类，还有name，address的属性

class Customer : public User {

private:

string name;

string address;

public:

Customer(string username, string password, string name, string address)

: User(username, password) {

this->name = name;

this->address = address;

}

string getName() {

return name;

}

string getAddress() {

return address;

}

};

//管理员

class Admin : public User {

public:

Admin(string username, string password)

: User(username, password) {}

};

//产品类

class Product {

public:

string name;

double price;

int num;//商品名，价格，数量

Product(string name, double price,int num) {

this->name = name;

this->price = price;

this->num = num;

}

};

//购物商城，可以展示商品内容，购买商品

class shoppingmall {

public:

vector<Product> products;//用于储存商城里的商品

void addItem(Product item) {//向向量里添加商品

int falg = 0;

for (int i = 0; i < products.size(); i++) {

if (products[i].name == item.name) {

products[i].num++;

falg = 1;

}

}

if (!falg) {

products.push\_back(item);//push\_back() 是 C++ 中 vector 容器的成员函数，用于在容器的末尾添加一个元素。

products[products.size() - 1].num++;

}

}

int searchProductByName(const string& name) const {//直接查找某商品

bool found = false;

//遍历向量

for (const auto& product : products) {

if (product.name == name) {

cout << "Name: " << product.name << endl;

cout << "Price: " << product.price << endl;

cout << "Num: " << product.num << endl;

found = true;//更改状态为已找到

return 1;

break;

}

}

if (!found) {

cout << "Product not found." << endl;//否则，未找到

return 0;

}

}

//根据关键词搜索并推荐

void recommendProductByKeyword(const string& keyword) const {

vector<Product> recommendedProducts;

for (const auto& product : products) {

if (product.name.find(keyword) != string::npos) {

recommendedProducts.push\_back(product);

}

}

//推荐向量为空

if (recommendedProducts.empty()) {

cout << "No recommended products found." << endl;

}

else {

cout << "Recommended Products:" << endl;

for (const auto& product : recommendedProducts) {

cout << "Name: " << product.name << endl;

cout << "Price: " << product.price <<endl;

cout << "Num: " << product.num << endl;

}

}

}

int return\_num() {//返回购物商城商品品类数

return products.size();

}

void removeItem(int index) {//移除商品

if (index >= 0 && index < products.size()) {

products.erase(products.begin() + index);

}

}

void printCart() {//打印购物车

cout << "Contents:" << endl;

for (int i = 0; i < products.size(); i++) {

cout << i + 1 << ". " << products[i].name << " - $"

<< products[i].price << " 库存: "<< products[i].num <<endl;

}

}

void resise\_name(int i,string name) {//修改商品名字

products[i].name = name;

}

void resise\_num(int i, int num) {//修改商品数量

products[i].num = num;

}

void resise\_pri(int i, double price) {//修改商品价格

products[i].price = price;

}

void buysomething(int i) {//购买商品后数量减少

products[i].num--;

}

int return\_pronum(int i) {//返回商品库存

return products[i].num;

}

Product return\_p(int i) {//返回Product

return products[i];

}

void save(const string& filename, Product i) {//将商品信息添加进文件

ofstream file(filename, ios::app);

if (file.is\_open()) {

file << i.name << "," << i.price << "," << i.num << "\n";

file.close();

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

void restoreFromDisk(const string& filename) {//将商品信息从文件中导至向量

ifstream file(filename);

products.clear();//先对向量内容进行清空

if (file.is\_open()) {

string line;

while (getline(file, line)) {//按行读取内容

string name;

double price;

int num;

size\_t pos = line.find(",");//寻找以，分割的第一个位置

if (pos != string::npos) {

name = line.substr(0, pos);

price = stod(line.substr(pos + 1));

pos = line.find(",", pos + 1);//第2个逗号

num = stoi(line.substr(pos + 1));

products.push\_back(Product(name, price, num));

}

}

file.close();

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

};

struct PurchaseHistory {//购买历史，包括商品以及购买时间

vector<Product> products;

string purchasetime;

};

//购物车，可以增删物品，获取物品总价格，以及展示商品

class ShoppingCart{

public:

double discountRate=0;

vector<Product> shproducts;

unordered\_map<string, double> coupons;//优惠券

vector<PurchaseHistory> purchaseHistories;

void init\_pro() {//初始化，使加入购物车的商品数量为1

for (int i = 0; i < shproducts.size(); i++) {

shproducts[i].num = 1;

}

}

Product return\_p(int i) {

return shproducts[i];

}

//erase()用于删除从begin开始后的index的物品

void removeItem(int index) {

if (index >= 0 && index < shproducts.size()) {

shproducts.erase(shproducts.begin() + index);

}

}

void addItem(Product& product) {

int falg = 0;

for (int i = 0; i < shproducts.size(); i++) {

if (shproducts[i].name == product.name) {

shproducts[i].num++;

falg = 1;

}

}

if (!falg) {

shproducts.push\_back(product);//push\_back() 是 C++ 中 vector 容器的成员函数，用于在容器的末尾添加一个元素。

shproducts[shproducts.size()-1].num++;

}

}

double buysomething(int i) {//购买商品返回其总价格

double pri = shproducts[i].price \* shproducts[i].num;

return pri;

}

void revise\_item(int index,int num) {//修改数量

shproducts[index].num=num;

}

int return\_num() {//返回购物车商品数量

return shproducts.size();

}

double getTotalPrice() {//计算总价

double totalPrice = 0;

for (int i = 0; i < shproducts.size(); i++) {

totalPrice += shproducts[i].price\* shproducts[i].num;

}

return totalPrice;

}

void printCart() {//打印购物车内容

cout << "Shopping Cart Contents:" << endl;

for (int i = 0; i < shproducts.size(); i++) {

cout << i + 1 << ". " << shproducts[i].name <<" 数量：" << shproducts[i].num << "件 "

<< " - $" << shproducts[i].price << endl;

}

cout << "Total Price: $" << getTotalPrice() << endl;

}

void addCoupon(const string& couponCode, double discount) {//添加优惠券以及对应优惠

coupons[couponCode] = discount;

}

void setDiscountRate(double rate) {

discountRate = rate;

}

double calculateTotalPriceAfterCoupon(double totalPrice, const string& couponCode) const {

auto itr = coupons.find(couponCode);//计算打完折后的总价钱

if (itr != coupons.end()) {

double discount = itr->second;

return totalPrice \* (1 - discount)\*(1-discountRate);

}

else {

return totalPrice \* (1 - discountRate);

}

}

void saveToDisk(const string& filename,Product i) {//将商品信息追加到文件里

ofstream file(filename, ios::app);

if (file.is\_open()) {

file << i.name << "," << i.price << "," << 1<< "\n";

file.close();

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

void savepurchase(const string& filename, Product i,string time) {

ofstream file(filename, ios::app);//将购买记录储存到文件里

if (file.is\_open()) {

file << i.name << "," << i.price << "," <<i.num<<"," << time << "\n";

file.close();

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

void restoreFromDisk(const string& filename) {

ifstream file(filename);//将文件信息导入至向量里

shproducts.clear();

if (file.is\_open()) {

string line;

while (getline(file, line)) {

string name;

double price;

int num;

size\_t pos = line.find(",");//以，分割

if(pos != string::npos) {

name = line.substr(0, pos);

price =stod(line.substr(pos + 1));

pos = line.find(",",pos+1);//第二个，

num= stoi(line.substr(pos +1));

shproducts.push\_back(Product(name, price,num));

}

}

file.close();//关闭文件

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

void restorepurchase(const string& filename) {

ifstream file(filename);

purchaseHistories.clear();

if (file.is\_open()) {

string line;

while (getline(file, line)) {//按行读取

string name;

double price;

string time;

int num;

size\_t pos = line.find(",");//以，进行分割

if (pos != string::npos) {

name = line.substr(0, pos);

price = stod(line.substr(pos + 1));

pos = line.find(",", pos + 1);

num = stoi(line.substr(pos + 1));

pos = line.find(",", pos + 1);//第2个，

time = line.substr(pos + 1);

PurchaseHistory purchaseHistory;

vector<Product> temp;

temp.push\_back(Product(name, price, num));

purchaseHistory.products = temp;

purchaseHistory.purchasetime = time;

purchaseHistories.push\_back(purchaseHistory);//把包含产品购买时间和商品传入

}

}

file.close();

}

else {

cerr << "无法打开文件 " << filename << endl;

}

}

vector<Product> temp(int i) {//返回向量

vector<Product> temp;

temp.clear();

temp.push\_back(shproducts[i]);

return temp;

}

// 添加购买记录

void addPurchaseRecord(const vector<Product>& products, const string& purchaseTime) {

PurchaseHistory purchaseHistory;

purchaseHistory.products = products;

purchaseHistory.purchasetime = purchaseTime;

purchaseHistories.push\_back(purchaseHistory);

}

// 显示购物历史记录

void displayPurchaseHistory() {

if (purchaseHistories.empty()!=true) {

for (const auto& purchaseHistory : purchaseHistories) {

cout << "购买时间：" << purchaseHistory.purchasetime << endl;

for (const auto& product : purchaseHistory.products) {

cout << "商品名称：" << product.name;

cout << "，商品价格：" << product.price << endl;

}

cout << endl;

}

}

else

{

cout << "您并没有购买记录。" << endl;

}

}

// 根据时间展示购物数据分析

void displayPurchaseAnalysis(const string& purchaseTime) {

map<string, double> categoryAmounts;

if (!purchaseHistories.empty()) {//如果为不为空的话才能操作

for (const auto& purchaseHistory : purchaseHistories) {

if (purchaseHistory.purchasetime == purchaseTime) {

for (const auto& product : purchaseHistory.products) {

categoryAmounts[product.name] += product.price;

}

}

}

cout << "购买时间：" << purchaseTime << std::endl;

for (const auto& entry : categoryAmounts) {

std::cout << "商品名称：" << entry.first;

std::cout << "，购买金额：" << entry.second << std::endl;

}

}

else {

cout << "您并没有购买数据。" << endl;

}

}

};

int main() {

Admin admin("admin", "admin123");

vector<Customer> customers;

// 注册顾客

string username, password,name,address="null";

User\* currentUser = nullptr;

// 模拟购买过程

Product item1("iPhone-phone", 5999.99,1000);

Product item2("ASUS-Laptop", 6499.99,1000);

Product item3("daiannaflower", 60.99,50);

Product item4("cabuqinuoflower", 80.99, 50);

//添加商品

shoppingmall cart;

cart.addItem(item1);

cart.addItem(item2);

cart.addItem(item3);

cart.addItem(item4);

// 给顾客随机派送优惠券

random\_device rd;

mt19937 gen(rd());

uniform\_real\_distribution<double> dis(0.0, 0.2);

ShoppingCart content;

string couponCode = "I Love XD";

double couponValue = dis(gen); // 随机生成优惠券面值

content.addCoupon(couponCode, couponValue);

//活动优惠

double discountRate = 0.2; // 假设活动折扣为 20%

content.setDiscountRate(discountRate);

int pnum;

string filename;

string hisname;

//封面

int temp=-1;

for (int i = 0;i < 5;i++) cout << endl;

cout << " \*\*\* \*\*\* " << " \*\*\* \*\*\* " << endl;

cout << " \*\*\*\*\*\* \*\*\*\*\*\* " << " \*\*\*\*\*\* \*\*\*\*\*\* " << endl;

cout << " \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*" << " \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*" << endl;

cout << " \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\* " << " \*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\* " << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* " << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* " << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\* " << " \*\*\*\*\*\*\*\*\*\*\* " << endl;

cout << " \*\*\*\*\*\*\* " << " \*\*\*\*\*\*\* " << endl;

cout << " \*\*\* " << " \*\*\* " << endl;

cout << " 太好逛了吧 " << " 太好逛了吧 " << endl;

cout << endl;

//菜单选择

while (temp) {

switch (temp) {

case -2: {//管理员功能

int choise;

string name;

double price;

int num=0;

int list;

int flag = 1;

string mallfile;

while (flag) {

cout << "选择1.上架物品 2.修改信息 3.删除物品 4.退出 ：";

cin >> choise;

cart.printCart();

switch (choise) {

case 1:{//上架物品

mallfile = "mall\_item.txt";

ofstream input(mallfile, ios::app);//追加的形式添加

if (input.is\_open()) {

cout << "上架物品。输入商品名 价格 库存数量：";

cin >> name >> price >> num;

Product item(name, price, num);

cart.addItem(item);//添置

cart.save(mallfile, item);//加入文件

cart.restoreFromDisk(mallfile);//导入至向量

input << name << " " << price << " " << num << endl;

input.close();

}

else {

cout << "无法打开文件" << endl;

}

break;

}

case 2: {//修改

string filename = "mall\_item.txt"; // 文件名

string tempFilename = "temp.txt"; // 临时文件名

int lineIndex; // 需要修改的行号，从0开始

int elementIndex; // 需要修改的元素在行中的位置，从0开始

ifstream input(filename); // 打开源文件读取内容

ofstream tempOutput(tempFilename); // 打开临时文件准备写入内容

if (!input.is\_open() || !tempOutput.is\_open()) {

cout << "打开文件失败。" << endl;

return 0;

}

string line;

cout << "修改信息。输入要修改物品序号：";

cin >> list;

lineIndex = list - 1;

int currentLineIndex = 0;

while (getline(input, line)) {

if (currentLineIndex == lineIndex) {

stringstream ss(line);

string element;

vector<string> elements;

int currentElementIndex = 0;

cout << "修改 1.商品名 2.价格 3.库存数量 ";

cin >> choise;

while (ss >> element) {//以空格分割直接遍历读取

elements.push\_back(element);

}

if (choise == 1) {

cout << "请输入修改后的商品名：";

cin >> name;

cart.resise\_name(lineIndex, name);//把行号和需要修改的传入

elements[0] = name;//修改name

}

else if (choise == 2) {

cout << "请输入修改后的价格：";

cin >> price;

cart.resise\_pri(lineIndex, price);//把行号和需要修改的传入

elements[1] = to\_string(price);//修改price

}

else if (choise == 3) {

cout << "请输入修改后的库存：";

cin >> num;

cart.resise\_num(lineIndex, num);//把行号和需要修改的传入

elements[2] = to\_string(num);//修改num

}

for (int i = 0;i < 3;i++) {

tempOutput << element[i] << " ";

}//只更正需要修改的那个，其余的照常加入

}

else {//如果不是需要修正的行，照常添加进临时文件中

tempOutput << line <<endl;

}

currentLineIndex++;//继续遍历

}

input.close();

tempOutput.close();

remove(filename.c\_str()); // 删除原始文件

rename(tempFilename.c\_str(), filename.c\_str()); // 重命名临时文件为原始文件的名称

break;

}

case 3: {

cout << "删除物品。输入要删除物品序号：";

cin >> list;

cart.removeItem(list);

string filename = "mall\_item.txt"; // 文件名

string tempFilename = "temp.txt"; // 临时文件名

int lineIndex; // 需要修改的行号，从0开始

ifstream input(filename); // 打开源文件读取内容

ofstream tempOutput(tempFilename); // 打开临时文件准备写入内容

if (!input.is\_open() || !tempOutput.is\_open()) {

cout << "打开文件失败。" << endl;

return 0;

}

string line;

lineIndex = list - 1;

int currentLineIndex = 0;

while (getline(input, line)) {//按行遍历

if (currentLineIndex != lineIndex) {

tempOutput << line << endl;

}

currentLineIndex++;

}

input.close();

tempOutput.close();

remove(filename.c\_str()); // 删除原始文件

rename(tempFilename.c\_str(), filename.c\_str()); // 重命名临时文件为原始文件的名称

break;

}

case 4: {

flag = 0;

temp = -1;

break;

}

}

}

break;

}

case -1: {

//打印主菜单

cout << " ==============================================================================================" << endl;

cout << " 菜单" << endl;

cout << " ==============================================================================================" << endl;

// 设置列宽和对齐方式

cout << " | " << setw(4) << left << "选项"

<< " | " << setw(73) << left << " 功能" << " |" << endl;

cout << " ----------------------------------------------------------------------------------------------" << endl;

cout << " | " << setw(4) << left << " 1"

<< " | " << setw(73) << left << "选项一 登录账号" << " |" << endl;

cout << " | " << setw(4) << left << " 2"

<< " | " << setw(73) << left << "选项二 注册账号" << " |" << endl;

cout << " | " << setw(4) << ::left << " 3"

<< " | " << setw(73) << std::left << "选项三 进入购物商场" << " |" << endl;

cout << " | " << setw(4) << left << " 4"

<< " | " << setw(73) << left << "选项四 查看购物车" << " |" << endl;

cout << " | " << setw(4) << left << " 5"

<< " | " << setw(73) << left << "选项五 用户管理" << " |" << endl;

cout << " | " << setw(4) << left << " 6"

<< " | " << setw(73) << left << "选项六 查询商品" << " |" << endl;

cout << " =============================================================================================" << endl;

cout << " 请输入您的选择：";

cin >> temp;

system("cls");

break;

}

//登录界面

case 1: {

system("cls");

cout << "登录中......." << endl;

cout << "Username: ";

cin >> username;

cout << "Password: ";

cin >> password;

int flag = 0;

//管理员登录

if (username == admin.getUsername() && password == admin.getPassword()) {

currentUser = &admin;

cout << "Admin login successful" << endl;

temp = -2;

}

//遍历user文件，登录用户账号

else {

int i;

string line;

ifstream inputfile("user\_inf.txt");

while (getline(inputfile, line)) {

istringstream iss(line);

vector<string> elements;

string element;

while (iss >> element) {

elements.push\_back(element);

}

if (elements[0] == username && elements[1] == password) {

cout << "成功登录" << endl;

temp = 3;

static Customer customer(elements[0], elements[1], elements[2], elements[3]);

customers.push\_back(customer);

currentUser = &customer;

filename = currentUser->getUsername() + ".txt";//以用户名命名购物车文件

hisname = currentUser->getUsername() + "pur.txt";//以用户名命名购物历史文件

content.restoreFromDisk(filename);//将文件内容导入向量

system("cls");

}

}//如果用户不在文件中，则转至注册界面

if(temp!=3) {

char ch;

cout << "对不起，您好像还没有注册。请问需要注册账号么？（y/n）："<<endl;

cin >> ch;

if (ch == 'y' || ch == 'Y') {

temp = 2;//注册选项

system("cls");//清屏

}

}

}

break;

}//注册界面

case 2: {

cout << "注册账户中......" << endl;

ofstream input("user\_inf.txt",ios::app);

cout << "Username: ";

cin >> username;

cout << "Password: ";

cin >> password;

cout << "name: ";

cin >> name;

cout << "address: ";

cin >> address;

if (input.is\_open()) {

input << username << " " << password << " " << name << " " << address<<endl;

input.close();

}

else {

cout << "无法打开文件" << endl;

}

static Customer customer(username, password, name, address);

customers.push\_back(customer);

currentUser = &customer;

system("cls");

cout << "恭喜您注册成功！！";

temp = 3;//自动转至购物商城界面

break;

}

//购物商城

case 3: {

char ch;

string filename = currentUser->getUsername() + ".txt";

cout << "Welcome, " << static\_cast<Customer\*>(currentUser)->getName() << "!!!" << endl;

cout << "Address: " << static\_cast<Customer\*>(currentUser)->getAddress() << endl;

cart.printCart();//打印购物商城内容

cout << "今天优惠大促，全场优惠" << content.discountRate \* 100 << '%' << "并享受随机金额优惠卷叠加使用！"

<< "走过路过，不要错过！" << endl;

string checkpass;

int a=0;

cout << "请问您有心仪的物品么？（y/n）： ";

cin >> ch;

if (ch == 'y' || ch == 'Y') {

if (currentUser == &admin) {

cout << "Only customers can buy products." << endl;//管理员无法操作

temp = -2;

}

else {

while (1) {

if((a++)) cart.printCart();//只有第一次进来的时候不打印，

cout << "请输入物品序号： ";

cin >> pnum;

if (pnum <= cart.return\_num()) {

system("cls");

cout << "加入购物车成功!" << endl;

Product a = cart.return\_p(pnum - 1);

content.saveToDisk(filename, a);//将商品添加至文件中

content.restoreFromDisk(filename);//并将文件内容导入向量

content.init\_pro();

break;

}

else {

system("cls");

cout << "输入序号有误,请重新输入" << endl;

}

}

}

}

else {

system("cls");

temp = -1;

}

break;

}

//查看购物车

case 4: {

if(currentUser== &admin) {

cout << "Only customers can check their shoppingcar" << endl;

temp = -2;

}

else if (currentUser == NULL) {

cout << "不好意思，您还没有登录。" << endl;

temp = 1;

}

else{

int t=1;

int a,num;

char ch;

string checkpass;

string inputCouponCode;

while (t!=4) {

system("cls");

time\_t currentTime = time(nullptr);

// 格式化时间戳为字符串

char buffer[80];

strftime(buffer, sizeof(buffer), "%Y-%m-%d", std::localtime(&currentTime));

cout << "购物车(" << content.return\_num() << ")" << endl;

// 打印购物车内容和总价

content.printCart();

cout << "1.购买 2.修改 3.删除 4.返回主界面:";

cin >> t;

switch (t) {

case 1: {

cout << "请输入物品序号： ";

cin >> a;

//输入密码才能正常购买

while (1) {

cout << "请输入您的账户密码以保证正常支付： ";

cin >> checkpass;

cout << "请问您有要使用的优惠券嘛？(y/n):";

cin >> ch;

if (ch == 'y' || ch == 'Y') {

//输入优惠券码

cout << "Enter your coupon code: ";

cin.ignore();

getline(cin,inputCouponCode);

if (checkpass == currentUser->getPassword()&& inputCouponCode== couponCode) {

content.addPurchaseRecord(content.temp(a - 1), buffer);

double num = content.buysomething(a - 1);

double calnum = content.calculateTotalPriceAfterCoupon(num, inputCouponCode);

cout << "恭喜您支付成功！！原价为：" << num << "元 "<<"折后价为："<<calnum<<"元" << endl;

content.savepurchase(hisname, content.return\_p(a - 1), buffer);

content.removeItem(a - 1); //移除购物车中已购买的商品

break;

}

else if(checkpass != currentUser->getPassword()&& inputCouponCode == couponCode) {

cout << "密码错误，请重新输入。" << endl;

}

else if (inputCouponCode != couponCode&& checkpass == currentUser->getPassword()) {

cout << "优惠码错误，请重新输入。" << endl;

}

}

else {

if (checkpass == currentUser->getPassword()) {

content.addPurchaseRecord(content.temp(a - 1), buffer);

cout << "恭喜您支付成功！！支付价为：" << content.buysomething(a - 1) << "元"<< endl;

content.savepurchase(hisname, content.return\_p(a - 1), buffer);

content.removeItem(a - 1);

break;

}

else {

cout << "密码错误，请重新输入。" << endl;

}

}

}

break;}

case 2: {

cout << "修改商品序号为：";

cin >> a;

cout << "数量更改为： ";

cin >> num;

content.revise\_item(a - 1, num);

string tempFilename = "temp.txt"; // 临时文件名

int lineIndex; // 需要修改的行号，从0开始

int elementIndex; // 需要修改的元素在行中的位置，从0开始

ifstream input(filename); // 打开源文件读取内容

ofstream tempOutput(tempFilename); // 打开临时文件准备写入内容

if (!input.is\_open() || !tempOutput.is\_open()) {

cout << "打开文件失败。" << endl;

return 0;

}

string line;

lineIndex = a - 1;

int currentLineIndex = 0;

while (getline(input, line)) {

if (currentLineIndex == lineIndex) {

stringstream ss(line);

string element;

vector<string> elements;

int currentElementIndex = 0;

while (getline(ss, element, ',')) {

elements.push\_back(element);

}

elements[2] = to\_string(num);

for (int i = 0;i < 3;i++) {

if (i != 2) tempOutput << elements[i] << ",";

else tempOutput << elements[i]<<endl;

}

}

else {

tempOutput << line << endl;

}

currentLineIndex++;

}

input.close();

tempOutput.close();

remove(filename.c\_str()); // 删除原始文件

rename(tempFilename.c\_str(), filename.c\_str()); // 重命名临时文件为原始文件的名称

system("cls");

cout << "页面刷新中，这是你新的购物车。" << endl;

cout << "购物车(" << content.return\_num() << ")" << endl;

content.printCart();

break;

}

case 3: {

cout << "移除商品序号为：";

cin >> a;

content.removeItem(a - 1);

string tempFilename = "temp.txt"; // 临时文件名

int lineIndex; // 需要修改的行号，从0开始

ifstream input(filename); // 打开源文件读取内容

ofstream tempOutput(tempFilename); // 打开临时文件准备写入内容

if (!input.is\_open() || !tempOutput.is\_open()) {

cout << "打开文件失败。" << endl;

return 0;

}

string line;

lineIndex = a - 1;

int currentLineIndex = 0;

while (getline(input, line)) {

if (currentLineIndex != lineIndex) {

tempOutput << line << endl;

}

currentLineIndex++;

}

input.close();

tempOutput.close();

remove(filename.c\_str()); // 删除原始文件

rename(tempFilename.c\_str(), filename.c\_str()); // 重命名临时文件为原始文件的名称

system("cls");

cout << "页面刷新中，这是你新的购物车。" << endl;

cout << "购物车(" << content.return\_num() << ")" << endl;

content.printCart();

break;

}

case 4: {

temp = -1;

system("cls");

break;

}

}

}

}

break;

}

//用户管理，可以修改用户密码

case 5: {

string newPassword;

int choice;

cout << "1.修改购物密码 2.查看购物历史 3.获取购物数据分析 4.返回主菜单: ";

if (currentUser == NULL) {

cout << "不好意思，您还没有登录。" << endl;

temp = 1;

break;

}

else {

cin >> choice;

}

switch (choice) {

case 1: {

/\*修改顾客密码\*/

if (Customer\* currentCustomer = static\_cast<Customer\*>(currentUser)) {

cout << "Enter new password: ";

cin >> newPassword;

currentCustomer->setPassword(newPassword);

cout << "Password updated." << endl;

}

else {

temp = 1;

}

string line;

string filename = "user\_inf.txt";

string tempFilename = "temp.txt"; // 临时文件名

ifstream input(filename); // 打开源文件读取内容

ofstream tempOutput(tempFilename); // 打开临时文件准备写入内容

if (!input.is\_open() || !tempOutput.is\_open()) {

cout << "打开文件失败。" << endl;

return 0;

}

while (getline(input, line)) {

istringstream iss(line);

vector<string> elements;

string element;

while (iss >> element) {

elements.push\_back(element);

}

if (elements[0] == username && elements[1] == password) {

elements[1] = newPassword;

}

}

tempOutput << input.rdbuf();//通过调用rdbuf()函数可以获得streambuf的指针，

//进而可以直接对缓冲区进行操作，如读取或写入数据，跳转位置等

input.close();

tempOutput.close();

remove(filename.c\_str()); // 删除原始文件

rename(tempFilename.c\_str(), filename.c\_str()); // 重命名临时文件为原始文件的名称

break;

}

case 2: {//展示购物历史

content.restorepurchase(hisname);

content.displayPurchaseHistory();

break;

}

case 3: {//查询一定时间下的购物数据，并分析

string time;

content.restorepurchase(hisname);

cout << "请输入你想查询的日期,格式为年-月-日: " ;

cin >> time;

content.displayPurchaseAnalysis(time);

break;

}

case 4: {

temp = -1;

break;

}

}

break;

}

case 6: {

string searchName;

string recommendKeyword;

int falg = 0;

// 查询商品信息

cout << "Enter the name of the product to search: ";

cin >> searchName;

falg=cart.searchProductByName(searchName);

if (!falg) {//查询关键信息

cout << "Enter a keyword for product recommendation: ";

cin >> recommendKeyword;

cart.recommendProductByKeyword(recommendKeyword);

}

break;

}

default: {

cout << "输入错误，请重新输入"<<endl;

temp = -1;

}

}

}

return 0;

}