

# 《程序设计基础课程设计》报告

(2023/2024 学年第二学期)

题目： 任务 8：商店销售管理系统

学号： 2023329600143

姓名： 唐韩宇

班级： 23 电子信息类 5 班

指导教师： 陶灵兵

浙江理工大学计算机学院

## 一、需求分析

设计并实现一个商品销售管理系统。商品信息包括商品编号、商品名称、进货价格、销售价格、商品库存等。

### 功能：

- (1) 系统采用菜单工作方式，保证用户界面清晰、友好、美观、易用。
- (2) 商品信息导入功能，可以从磁盘文件导入商品信息，支持 csv。
- (3) 提供商品信息的查看功能，可以输出所有商品的详细信息。
- (4) 提供商品查询功能，允许按类型或商品名称进行搜索，并将结果显示出来。
- (5) 新增商品信息功能，可以向商品库中添加新的商品信息。
- (6) 进货功能，进货时填写商品编号和进货数量，商品库存随之增加。
- (7) 销售功能，顾客购买商品时，输入商品编号，系统会自动生成销售清单，并统计本次销售的总额，同时商品库存会有所减少。
- (8) 扩展功能 1：增加了用户管理身份系统，分为管理员和顾客，不同身份有不同的权限。
- (9) 扩展功能 2：数据保存功能，管理员可以将当前的商品信息以及最后的库存保存到磁盘文件中，方便下次使用时加载。

## 二、总体设计

本商店销售管理系统功能模块如图 1 所示，共包含 2 个大模块：用户登录模块和数据操作模块。数据操作模块中分为导入数据模块、商品显示模块、商品搜索模块、商品品类添加模块、添加库存模块和导出数据模块。为了提高程序设计效率，本系统采用单链表实现所有操作。

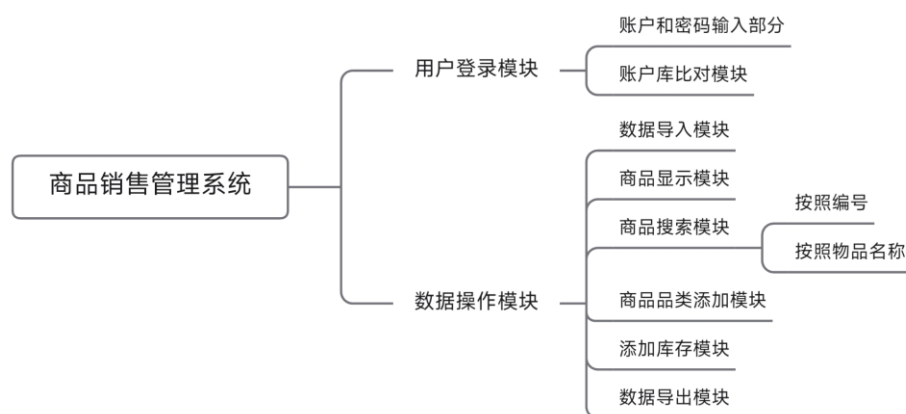


图 1 模块设计图

- (1) 账户登录界面，只有登录后才能进入系统，确保系统的安全性
- (2) 用户界面模块：采用菜单工作方式，以实现用户界面的清晰、友好、美观和易用性。
- (3) 商品信息导入模块：支持从磁盘文件（.csv 格式）导入商品信息。
- (4) 商品信息浏览模块：提供商品信息的完整查看，呈现所有商品的详细信息。
- (5) 商品查询模块：允许按类型或商品名称进行搜索，并将搜索结果进行展示。
- (6) 商品信息添加模块：用于向商品库中添加新的商品信息。
- (7) 进货模块：在进货时填写商品编号和进货数量，商品库存对应增加。
- (8) 销售模块：在顾客购买商品时，通过输入商品编号，系统会自动生成销售清单，并统计本次销售的总额，同时商品库存也会相应减少。
- (9) 数据保存模块：允许用户将当前的商品信息保存至磁盘文件中，方便下次使用时加载。

### 三、详细设计

#### (1) 系统总流程图

本系统的操作从人机交互界面的用户登录界面开始，用户应输入账号和用户名来进行登录，然后在 1-8 之间的数字选择要进行的操作，输入其他符号系统将提示未知信息的提示信息。具体流程图如图 2 所示：

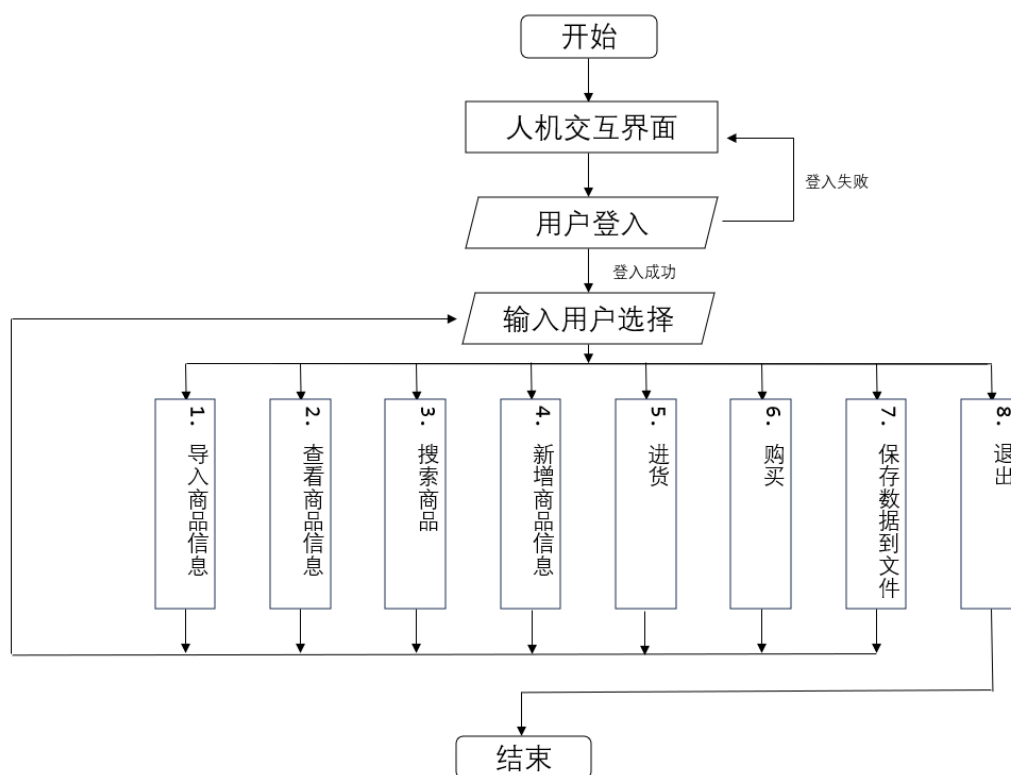


图 2 系统总流程图

### (2) 导入商品信息流程图

导入商品信息的操作，从打开 csv 文件开始，将每行数据读入，读入后通过 `isProductExists(tempID, tempName)`；函数来判断商品的 ID 或者 Name 是否已经存在，验证通过后才将数据导入，并输出“导入成功”字样，最后关闭文件。具体流程图如图 3 所示：

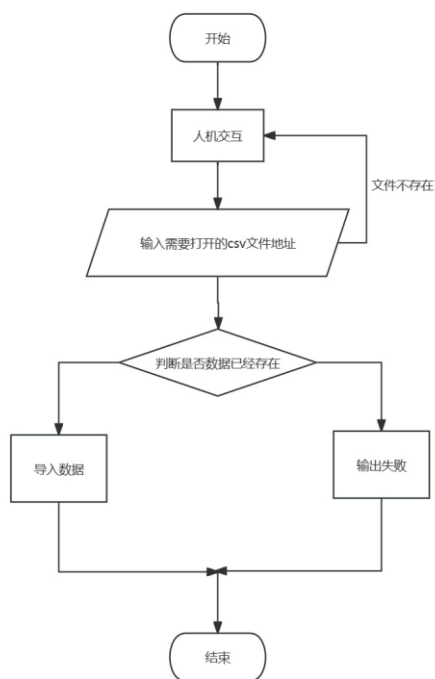


图 3 导入信息流程图

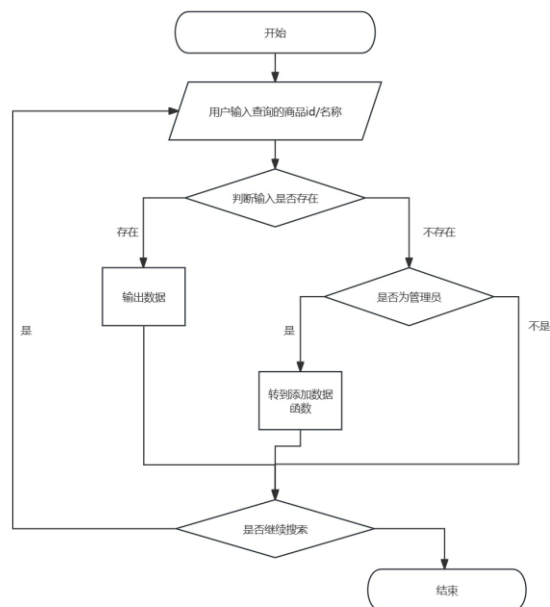


图 4 搜索商品流程图

### (3) 搜索商品流程图

搜索商品的操作，用户输入商品的编号和名称，然后在数据库中进行寻找，若找到，输出商品信息，没找到询问是否继续搜索。此函数传入了 `role` 这个身份参数，如果身份为管理员，会转到是否添加这个没有找到的商品到数据库中。具体流程图如图 4 所示：

### (4) 添加商品流程图

添加商品的操作，用户输入添加的商品的编号和名称，先通过 `isValidEnglishName()` 函数判定输入的名称是否全为英文，然后通过再次调用 `isProductExists(tempID, tempName)` 函数判断是否已经存在，若没有存在，依次从用户处获取进货价格，销售价格，库存量。具体流程图如图 5 所示。

### (5) 客户购买流程图

用户购买的流程，先显示所有商品，获取用户要购买物品的 ID 和数量，将用户的每次

购买存到一张表中，然后购买数量与库存进行比较，如果库存足够，输出购买成功，并且给出总价，询问用户是否结束购物，若结束，输出所有购买的商品，同时给出总额。具体流程图如图 6 所示。

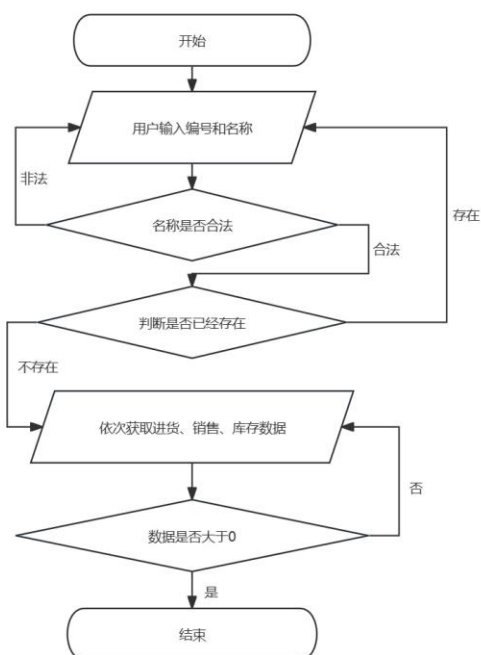


图 5 添加数据流程图

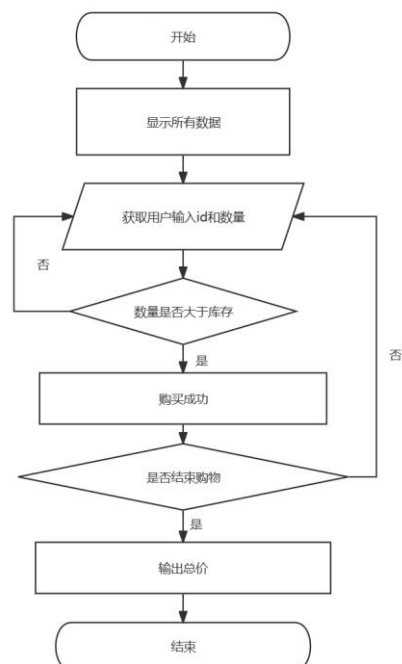


图 6 客户购买流程图

#### 四、 系统实现

本程序主要由 5 个文件构成：分别是主函数 main.c、有关数据操作的 Product.c、相应的头文件 Product.h、有关用户登录模块的 userAuthertication.c、相应的头文件 userAuthertication.h。文件还有一个 products.csv 用于文件的导入

##### 1、文件 myProgram.exe

myProgram.exe 与源程序位于同一目录下，以二进制文件的形式存储，是最后的输出结果。

##### 2、文件 main.c

```

//基础
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
//自定义头文件
#include "userAuthentication.h"
#include "Product.h"

// 功能函数
void importProductFromCSV(const char *filePath);// 从 CSV 文件导入商品信息
void viewProducts(); // 查看所有商品信息
void searchProduct(); // 按类型编号或商品名称搜索商品
void addProduct(); // 新增商品信息
void restockProduct(); // 进货
void sellProduct(); // 销售
void saveProductsToDisk(); // 数据保存

SetConsoleOutputCP(CP_UTF8);//中文显示

// 用户认证
UserRole role = userAuthentication();

int choice;
char filePath[256]; // 存储文件路径名称大小
char input[10];
char productId[10]; // 添加库存的编号
int additionalStock; // 添加库存的数量

while(1){
if (role != ROLE_NONE) {
    printf("登录成功. 您的角色是: %s\n", role == ROLE_ADMIN ? "管理员" : "用户");

    do {
        // 显示菜单
        printf("\n*****\n");
        printf("\n 商品销售管理系统\n");
        printf("1. 从文件导入商品信息\n");
        printf("2. 查看所有商品信息\n");
        printf("3. 搜索商品\n");
        printf("4. 新增商品信息\n");
    } while (1);
}
}

```

```

printf("5. 进货\n");
printf("6. 购买\n");
printf("7. 保存数据到文件\n");
printf("8. 退出\n");
printf("请输入您的选择: ");
printf("\n*****\n");
scanf("%d", &choice);
printf("\n*****\n");

switch(choice) {
    case 1:
        // 导入商品
        if (role == ROLE_ADMIN){
            printf("请输入文件路径(不需要加引号): ");
            scanf("%255s", filePath); // 接收用户输入的文件路径,限定了路径长度;
            importProductFromCSV(filePath);
        }
        else printf("权限不足.\n");

        break;

    case 2 :
        // 查看所有商品信息
        viewProducts();
        break;

    case 3:
        do {
            // 按类型编号或商品名称搜索商品
            searchProduct(role);

            // 搜索完成后询问用户是否需要继续搜索商品
            printf("需要继续搜索商品吗? (输入任意继续, 'no' 退出搜索): ");
            scanf("%9s", input);
            getchar(); // 获取并丢弃换行符
            // 将用户的输入转换为小写处理, 以防万一用户输入大写字符
            for(int i = 0; input[i]; i++) {
                input[i] = tolower(input[i]);
            }
        } while(strcmp(input, "no") != 0);
        break;

    case 4:
        // 实现新增商品信息逻辑

```

```

        if (role == ROLE_ADMIN){
            if (product_count < MAX_PRODUCTS) {
                addProduct();
            }
        }
        else {
            printf("商品库已满, 无法添加新的商品!\n");
        }
    }

    else printf("权限不足.\n");

    break;

case 5:
    // 实现进货逻辑
    if (role == ROLE_ADMIN){
        printf("请输入你想要增加库存的编号: ");
        scanf("%9s", productId); // 正确读取字符串, 最大长度为 9

        printf("请输入你想要增加库存的数量: ");
        scanf("%d", &additionalStock); // 正确读取整数

        restockProduct(productId, additionalStock); // 调用函数
    }

    else printf("权限不足.\n");

    break;

case 6:
    // 实现销售/购买逻辑
    viewProducts();
    sellProduct();
    break;

case 7:

    // 实现保存数据到文件逻辑
    saveProductsToDisk();
    break;

case 8:
    printf("退出程序.\n");
    exit(0); // 退出程序
    break;

```



```

        default:
            printf("未知选项，请重新输入。\\n");
        }
    } while (choice != 9);
}
else {
    printf("登录失败.\\n");
    printf("\\n*****\\n");
    printf("\\n 商品销售管理系统\\n");
    printf("1. 退出\\n");
    printf("2. 重新登录\\n");
    printf("请输入您的选择: ");
    printf("\\n*****\\n");
    scanf("%d", &choice);
    printf("\\n*****\\n");

    switch(choice){
        case 1:
            printf("退出程序.\\n");
            exit(0);
        case 2:
            role = userAuthentication();
        }
    }
}
return 0;
}

```

### 3、文件 userAuthertication.c

#### (1) 预处理:

```

//文件包含
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include "userAuthentication.h"

// 用户列表，假设先只有两个用户
User userDatabase[MAX_USERS] = {
    {"tanghanyu", "12345", ROLE_ADMIN},
    {"guke", "password", ROLE_USER},
};

```

```
// 用户认证函数实现，返回用户角色
UserRole userAuthentication() {
    char inputUsername[USERNAME_LEN];
    char inputPassword[PASSWORD_LEN];

    printf("请输入用户名: ");
    scanf("%s", inputUsername);
    printf("请输入密码: ");
    scanf("%s", inputPassword);

    for (int i = 0; i < MAX_USERS; ++i) {
        if (strcmp(userDatabase[i].username, inputUsername) == 0 &&
            strcmp(userDatabase[i].password, inputPassword) == 0) {
            return userDatabase[i].role; // 返回认证成功的用户角色
        }
    }
    return ROLE_NONE; // 认证失败返回无角色
}
```

#### 4、文件 userAuthertication.h

```
//文件包含
#include <stdbool.h>
```

##### (1) 数据类型定义：

```
// 定义用户角色
typedef enum { //枚举变量
    ROLE_NONE,
    ROLE_ADMIN,
    ROLE_USER
} UserRole;

// 用户登录结构
typedef struct {
    char username[USERNAME_LEN]; // 用户数组
    char password[PASSWORD_LEN]; // 密码数组
    UserRole role;                // 用户角色
} User;
```

##### (2) 全局变量定义和函数声明

```
// 用户认证函数声明，返回认证成功的用户角色
UserRole userAuthentication();
```

## 5、文件 Product.c 和 h 文件

### (1) 预处理

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h> // 为 isalpha 函数包含的头文件
#include "Product.h"
#include "userAuthentication.h"
```

### (2) 数据类型定义

```
typedef struct {
    char productID[10];
    char productName[50];
    float purchasingPrice;    // 进货价格
    float sellingPrice;       // 销售价格
    int stock;                // 商品库存
    float sales;              // 销售额
} Product; //用于存储商品的各种信息

typedef struct {
    char productName[50];
    int quantity;
    float totalPrice;
} PurchasedItem; //定义最后账单的结构体
int product_count = 0; //记录库存商品
int purchasedCount = 0; //记录已经购买的商品
```

### (3) 全局变量定义和函数声明

```
#define MAX_PRODUCTS 200 // 设定最大商品数量，全局变量在此处修改
#define MAX_PRODUCTS 200 // 设定最大商品数
#define MAX_PURCHASED 100 // 设定单次购买最大商品种类

extern Product products[MAX_PRODUCTS]; //在对数组进行处理
extern PurchasedItem purchasedItems[MAX_PURCHASED];
extern int product_count;

//其他函数
int findProductIndexById(const char* id);
int isProductExists(const char* productID, const char* productName);
int isValidEnglishName(const char *str);

// 有关 csv 文件的操作；
void importProductFromCSV(const char *filePath);
void viewProducts();
```

```

void searchProduct(role);
void addProduct();
void restockProduct(const char* productId, int additionalStock);
void saveProductsToDisk();

```

(4) 每个具体函数

findProductIndexById () //查找增加库存的编号是否存在

```

int findProductIndexById(const char* id) {
    for (int i = 0; i < product_count; ++i) {
        if (strcmp(products[i].productID, id) == 0) {
            return i;
        }
    }
    return -1;
}

```

isProductExists () // 检查商品编号或名称是否存在

```

int isProductExists(const char* productID, const char* productName) {
    for (int i = 0; i < product_count; i++) {
        if (strcmp(products[i].productID, productID) == 0 || strcmp(products[i].productName,
productName) == 0) {
            return 1; // 如果商品编号或名称已经存在, 返回 1
        }
    }
    return 0; // 商品编号和名称都是新的, 返回 0
}

```

isValidEnglishName () // 检查字符串是否全为英文字符

```

int isValidEnglishName(const char *str) {
    while (*str) {
        if (!isalpha((unsigned char)*str)) {
            return 0;
        }
        str++;
    }
    return 1;
}

```

importProductFromCSV () // 从 CSV 文件导入商品信息的函数实现

```

void importProductFromCSV(const char *filePath) {
    FILE *fp = fopen(filePath, "r");
    if (fp == NULL) {
        printf("无法打开文件。 \n");
        return;
    }

    char buffer[256];

```

```

fgets(buffer, 256, fp); // 读取并忽略首行

while (fgets(buffer, 256, fp)) {
    if (product_count >= MAX_PRODUCTS) {
        printf("已达到商品数量上限，无法添加更多商品。\\n");
        break; // 停止读取文件和添加新产品
    }

    char tempID[10];
    char tempName[50];
    float tempPurchasingPrice, tempSellingPrice;
    int tempStock;
    float tempSales;

    sscanf(buffer, "%[^,],%[^,],%f,%f,%d,%f",
           tempID,
           tempName,
           &tempPurchasingPrice,
           &tempSellingPrice,
           &tempStock,
           &tempSales);

    // 检查编号和名称的唯一性
    if (isProductExists(tempID, tempName)) {
        printf("导入失败：商品编号或名称已存在 - %s, %s\\n", tempID, tempName);
    } else {
        // 如果验证通过，则将读取的数据保存到数组中
        strcpy(products[product_count].productID, tempID);
        strcpy(products[product_count].productName, tempName);
        products[product_count].purchasingPrice = tempPurchasingPrice;
        products[product_count].sellingPrice = tempSellingPrice;
        products[product_count].stock = tempStock;
        products[product_count].sales = tempSales;

        printf("\\n 导入成功\\n");
        printf("Product ID: %s, Name: %s, Purchasing Price: %.2f, Selling Price: %.2f, Stock: %d,
Sales: %.2f\\n",
               products[product_count].productID,
               products[product_count].productName,
               products[product_count].purchasingPrice,
               products[product_count].sellingPrice,
               products[product_count].stock,
               products[product_count].sales);
    }
}

```

```

        product_count++; // 更新计数器
    }
}

fclose(fp); // 文件读取完毕，关闭文件
}

```

viewProducts() //实现数据展现

```

void viewProducts() {
    if (product_count == 0) {
        printf("没有商品信息可显示。 \n");
        return; // 如果没有商品，则直接返回
    }

    // 遍历并打印所有商品信息
    for (int i = 0; i < product_count; ++i) {
        printf("Product ID: %s, Name: %s, Purchasing Price: %.2f, Selling Price: %.2f, Stock: %d, Sales: %.2f\n",
               products[i].productID,
               products[i].productName,
               products[i].purchasingPrice,
               products[i].sellingPrice,
               products[i].stock,
               products[i].sales);
    }
}
}

```

searchProduct(role) //实现商品搜索

```

void searchProduct(role) {
    char searchKey[50];
    printf("请输入商品类型编号或商品名称进行搜索: ");
    scanf("%49s", searchKey);
    getchar(); // 去除换行字符

    int flag = 0; // 用于记录是否找到至少一个匹配的商品

    for (int i = 0; i < product_count; ++i) {
        // 使用 strcasecmp 函数来实现不区分大小写的比较
        if (strcasecmp(products[i].productID, searchKey) == 0 ||
            strcasecmp(products[i].productName, searchKey) == 0) {
            printf("找到商品: Product ID: %s, Name: %s, Purchasing Price: %.2f, Selling Price: %.2f,

```

```

Stock: %d, Sales: %.2f\n",
        products[i].productID,
        products[i].productName,
        products[i].purchasingPrice,
        products[i].sellingPrice,
        products[i].stock,
        products[i].sales);
    flag = 1;
}
}

if (!flag) {
    printf("没有找到匹配的商品。 \n");
    char choice;
    if (role == ROLE_ADMIN)
    {
        printf("是否需要添加一个新商品? (y/n): ");
        scanf("%c", &choice);
        getchar();
        if (choice == 'y' || choice == 'Y') {
            addProduct();
        } else {
            printf("没有添加新商品。 \n");
        }
    }
    else {
        //无操作
    }
}
}

```

addProduct() //实现商品添加

```

void addProduct() {

    char tempID[10];
    char tempName[50];
    int validInput; // 用于控制循环的变量
    int validName; //用于判断是否全是英文

    do {
        printf("请输入新商品的类型编号: ");
        scanf("%9s", tempID);
        getchar(); // 清理输入缓冲区
    } while (1);
}

```

```

do {
    printf("请输入新商品的名称 (必须为英文): ");
    scanf("%49s", tempName);

    // 清理输入缓冲区
    int c;
    while ((c = getchar()) != '\n' && c != EOF); // 清除换行或 EOF

    if (isValidEnglishName(tempName)) {
        validName = 1;
        printf("输入的名称有效: '%s'\n", tempName);
    } else {
        printf("商品名称必须全部为英文字符, 请重新输入.\n");
        validName = 0;
    }
} while (!validName);

// 检查编号和名称的唯一性
if (isProductExists(tempID, tempName)) {
    printf("该商品编号或名称已存在, 请重新添加商品信息.\n");
    validInput = 0; // 继续循环
} else {
    validInput = 1; // 退出循环, 进行下一步
}
} while (validInput == 0); // 若编号或名称存在, 提示用户重新输入

// 经过上面的循环, 现在得到了唯一的商品编号和名称
strcpy(products[product_count].productID, tempID);
strcpy(products[product_count].productName, tempName);

printf("请输入新商品的进货价格: ");
while(scanf("%f", &products[product_count].purchasingPrice) != 1 ||
products[product_count].purchasingPrice < 0) {
    while(getchar() != '\n');
    printf("无效输入, 请输入非负数作为商品的进货价格: ");
}

printf("请输入新商品的销售价格: ");
while(scanf("%f", &products[product_count].sellingPrice) != 1 || products[product_count].sellingPrice
< 0) {
    while(getchar() != '\n');
    printf("无效输入, 请输入非负数作为商品的销售价格: ");
}

```



```

    }

    printf("请输入新商品的库存量: ");
    while(scanf("%d", &products[product_count].stock) != 1 || products[product_count].stock < 0) {
        while(getchar() != '\n');
        printf("无效输入, 请输入非负数作为商品的库存量: ");
    }

    products[product_count].sales = 0;

    product_count++;

    printf("新商品已成功添加到商品库!\n");
}

```

restockProduct() //实现进货

```

void restockProduct(const char* productId, int additionalStock) {
    if (additionalStock < 0) {
        printf("进货数量不能为负数。 \n");
        return;
    }

    int index = findProductIndexById(productId);
    if (index == -1) {
        printf("没有找到 ID 为 %s 的商品。 \n", productId);
        return;
    }

    products[index].stock += additionalStock;
    printf("商品 \"%s\" 的库存已更新。新库存量: %d\n", products[index].productName,
products[index].stock);
}

```

sellProduct() //实现物品购买

```

void sellProduct() {
    char input[50]; // 用户输入的商品 ID 或名称
    int quantity; // 用户希望购买的数量
    float totalSales = 0; // 初始化总销售额为 0

    printf("请输入商品 ID 和所需数量 (例如: 001 3), 或输入 exit 退出: ");

    while (scanf("%49s", input) && strcmp(input, "exit") != 0) {
        if (scanf("%d", &quantity) != 1) {
            printf("输入有误, 请重新输入。 \n");
            while (getchar() != '\n'); // 清空输入缓冲区

```

```

        continue;
    }

    int found = 0;
    for (int i = 0; i < MAX_PRODUCTS; ++i) {
        if (strcmp(products[i].productID, input) == 0 || strcmp(products[i].productName, input) == 0)
        {
            found = 1;
            if (products[i].stock >= quantity) {
                products[i].stock -= quantity;
                float total = quantity * products[i].sellingPrice;
                products[i].sales += total;
                totalSales += total;

                // 记录购买详情
                strcpy(purchasedItems[purchasedCount].productName, products[i].productName);
                purchasedItems[purchasedCount].quantity = quantity;
                purchasedItems[purchasedCount].totalPrice = total;
                purchasedCount++;

                printf("购买成功! %d 个%s, 总价: %.2f\n", quantity, products[i].productName,
total);
            } else {
                printf("库存不足! %s 仅剩%d。 \n", products[i].productName, products[i].stock);
            }
            break;
        }
    }

    if (!found) {
        printf("未找到商品。 \n");
    }
    printf("请输入商品 ID 和所需数量 (例如: 001 3), 或输入 exit 退出: ");
}

// 在结束时输出购买的所有商品详情
printf("\n 您购买的商品总结如下: \n");
for (int i = 0; i < purchasedCount; ++i) {
    printf("%s x%d, 总价: %.2f\n", purchasedItems[i].productName, purchasedItems[i].quantity,
purchasedItems[i].totalPrice);
}
// 打印出所有商品的总销售额
printf("所有购买的商品总销售额为: %.2f\n 购买操作完成。 \n", totalSales);
}

```

saveProductsToDisk()//实现表格另存为

```
void saveProductsToDisk() {
    char fileName[256]; // 存储用户输入的文件名
    printf("请输入想要保存的 CSV 文件名（包括.csv 扩展名）:");
    scanf("%s", fileName); // 从用户获取文件名

    FILE *file = fopen(fileName, "w"); // 打开文件以供写入，如果文件已存在则覆盖
    if (file == NULL) {
        fprintf(stderr, "无法打开文件 %s\n", fileName);
        return;
    }

    // 写 CSV 的标题行
    fprintf(file, "Product ID,Product Name,Purchasing Price,Selling Price,Stock,Sales\n");

    // 遍历 products 数组并将每个 Product 的信息写入文件
    for (int i = 0; i < MAX_PRODUCTS; i++) {
        fprintf(file, "%s,%s,%.2f,%.2f,%d,%.2f\n",
            products[i].productID,
            products[i].productName,
            products[i].purchasingPrice,
            products[i].sellingPrice,
            products[i].stock,
            products[i].sales);
    }

    fclose(file); // 关闭文件
    printf("商品信息已保存到 %s\n", fileName);
}
```

## 五、 系统测试

### 1、人机界面

运行系统即可进入用户登录人机界面，可登入不同账号如图 7 所示。

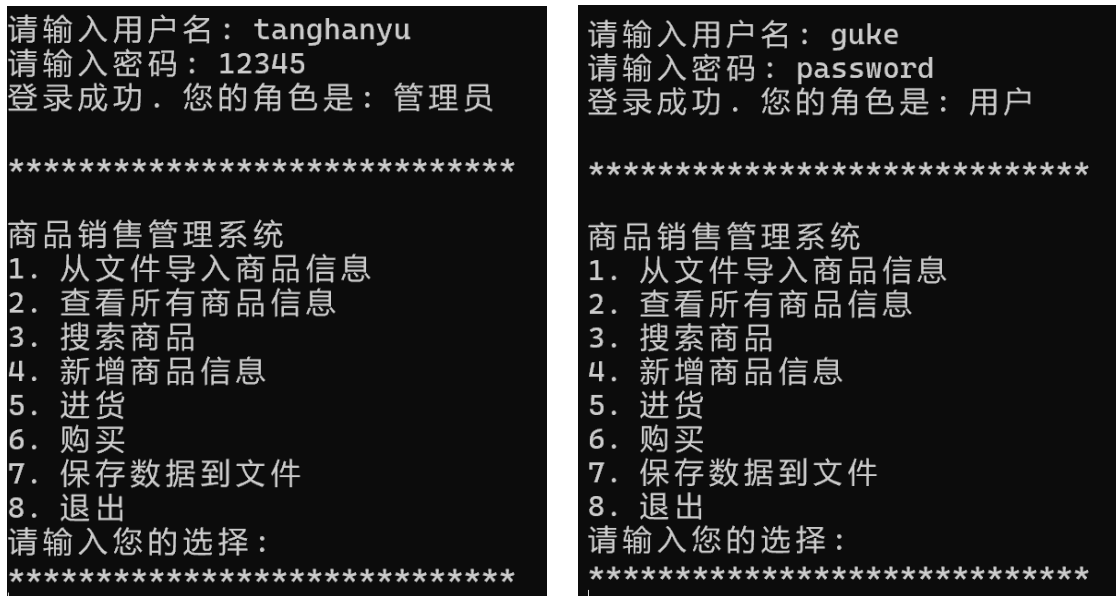


图 7 不同身份的登录图

如果输入的是未存储的账号或者密码错误，则会到重新登录界面。如图 8 所示。

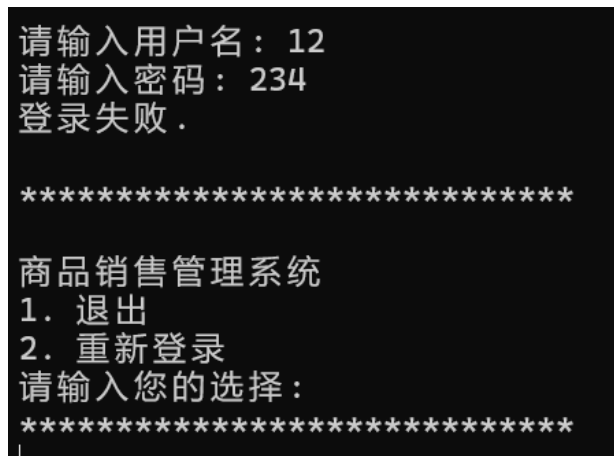


图 8 重新登录界面

## 2、信息导入

在主界面输入“1”即可进入导入信息提示，并且会对重复数据进行处理，如图 9、10 所示。

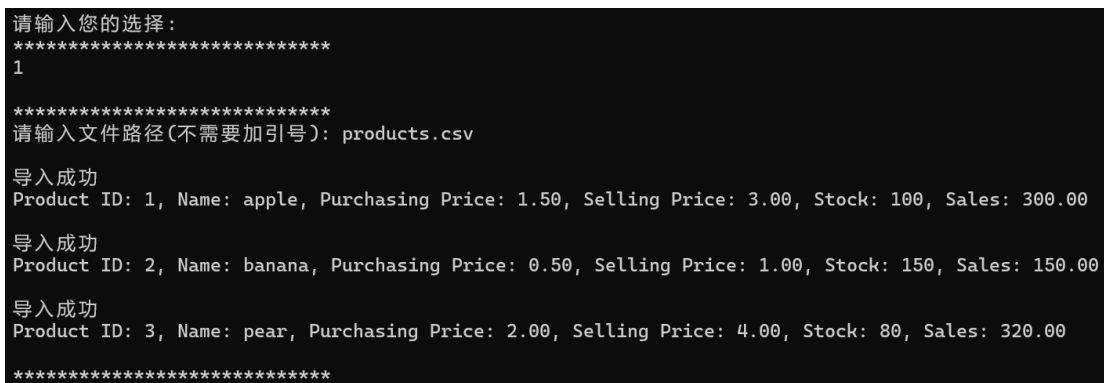


图 9 从文件导入商品信息界面图

```
请输入您的选择：
*****
1

*****
请输入文件路径(不需要加引号): products.csv
导入失败：商品编号或名称已存在 - 1, apple
导入失败：商品编号或名称已存在 - 2, banana
导入失败：商品编号或名称已存在 - 3, pear

*****
```

图 10 数据重复导入情况

## 2、查看信息

在主界面输入“2”即可进入查看所有信息，如图 11 所示。

```
请输入您的选择：
*****
2

*****
Product ID: 1, Name: apple, Purchasing Price: 1.50, Selling Price: 3.00, Stock: 100, Sales: 300.00
Product ID: 2, Name: banana, Purchasing Price: 0.50, Selling Price: 1.00, Stock: 150, Sales: 150.00
Product ID: 3, Name: pear, Purchasing Price: 2.00, Selling Price: 4.00, Stock: 80, Sales: 320.00
```

图 11 查看信息图

## 3、搜索物品

在主界面输入“3”即可进入查看所有信息，搜寻时，输入 ID 和名称都可以找到，而在未找到时给出，是否需要添加一个新的商品的提示。如图 12 所示。

```
请输入您的选择：
*****
3

*****
请输入商品类型编号或商品名称进行搜索：1
找到商品：Product ID: 1, Name: apple, Purchasing Price: 1.50, Selling Price: 3.00, Stock: 100, Sales: 300.00
需要继续搜索商品吗？(输入任意继续, 'no' 退出搜索): y
请输入商品类型编号或商品名称进行搜索：apple
找到商品：Product ID: 1, Name: apple, Purchasing Price: 1.50, Selling Price: 3.00, Stock: 100, Sales: 300.00
需要继续搜索商品吗？(输入任意继续, 'no' 退出搜索): yes
请输入商品类型编号或商品名称进行搜索：4
没有找到匹配的商品。
是否需要添加一个新商品？(y/n): y
请输入新商品的类型编号：4
请输入新商品的名称（必须为英文）：12
商品名称必须全部为英文字符，请重新输入。
请输入新商品的名称（必须为英文）：orange
输入的名称有效：'orange'
请输入新商品的进货价格：12
请输入新商品的销售价格：14
请输入新商品的库存量：10
新商品已成功添加到商品库！
需要继续搜索商品吗？(输入任意继续, 'no' 退出搜索): no
```

图 12 搜索商品图

#### 4、添加物品

在主界面输入“4”即可添加物品，如图 13 所示。

```
请输入您的选择：
*****
4

*****
请输入新商品的类型编号：5
请输入新商品的名称（必须为英文）：malen
输入的名称有效：'malen'
请输入新商品的进货价格：20
请输入新商品的销售价格：40
请输入新商品的库存量：50
新商品已成功添加到商品库！
```

图 13 添加商品图

#### 5、修改库存

在主界面输入“5”即可进入添加库存，添加时，如果没找到不会添加，而在找到时添加完后也会再一次显示库存量。如图 14、15 所示。

```
请输入您的选择：
*****
5

*****
请输入你想要增加库存的编号：6
请输入你想要增加库存的数量：12
没有找到ID为 6 的商品。
```

图 14 未找到时的图

```
请输入您的选择：
*****
5

*****
请输入你想要增加库存的编号：2
请输入你想要增加库存的数量：10
商品 "banana" 的库存已更新。新库存量：160

*****
```

图 15 找到时的图

6、购买物品

在主界面输入“6”即可购买，会和库存进行比较，库存也会实时更新，如图 16 所示。

```
请输入您的选择：
*****
6

*****
Product ID: 1, Name: apple, Purchasing Price: 1.50, Selling Price: 3.00, Stock: 100, Sales: 300.00
Product ID: 2, Name: banana, Purchasing Price: 0.50, Selling Price: 1.00, Stock: 160, Sales: 150.00
Product ID: 3, Name: pear, Purchasing Price: 2.00, Selling Price: 4.00, Stock: 80, Sales: 320.00
Product ID: 4, Name: orange, Purchasing Price: 12.00, Selling Price: 14.00, Stock: 10, Sales: 0.00
Product ID: 5, Name: malen, Purchasing Price: 20.00, Selling Price: 40.00, Stock: 50, Sales: 0.00
请输入商品ID和所需数量（例如：001 3），或输入exit退出：1 5
购买成功！5个apple，总价：15.00
请输入商品ID和所需数量（例如：001 3），或输入exit退出：2 5
购买成功！5个banana，总价：5.00
请输入商品ID和所需数量（例如：001 3），或输入exit退出：1 1000
库存不足！apple仅剩95.
请输入商品ID和所需数量（例如：001 3），或输入exit退出：exit

您购买的商品总结如下：
apple x5, 总价：15.00
banana x5, 总价：5.00
所有购买的商品总销售额为：20.00
购买操作完成。
```

图 16 购物时的图

7、数据导出

在主界面输入“7”即可导出数据，如图 17、18、19 所示。

```
请输入您的选择：
*****
7

*****
请输入想要保存的CSV文件名（包括.csv扩展名）：goumail.csv
商品信息已保存到 goumail.csv
```

图 17 保存数据图





|   |                 |                      |
|---|-----------------|----------------------|
|  .vscode       | 2024/5/14 20:27 | 文件夹                  |
|  goumail.csv   | 2024/5/15 14:17 | Microsoft Excel 逗... |
|  main.c        | 2024/5/15 11:01 | C 源文件                |
|  myProgram.exe | 2024/5/15 10:58 | 应用程序                 |

图 18 数据保存位置图

| A          | B            | C          | D             | E     | F     |  |
|------------|--------------|------------|---------------|-------|-------|--|
| Product ID | Product Name | Purchasing | Selling Price | Stock | Sales |  |
| 1          | apple        | 1.5        | 3             | 95    | 315   |  |
| 2          | banana       | 0.5        | 1             | 155   | 155   |  |
| 3          | pear         | 2          | 4             | 80    | 320   |  |
| 4          | orange       | 12         | 14            | 10    | 0     |  |
| 5          | malen        | 20         | 40            | 50    | 0     |  |
|            |              | 0          | 0             | 0     | 0     |  |
|            |              | 0          | 0             | 0     | 0     |  |
|            |              | 0          | 0             | 0     | 0     |  |
|            |              | 0          | 0             | 0     | 0     |  |
|            |              | 0          | 0             | 0     | 0     |  |

图 19 保存的数据的图

## 8、权限设置

当用户为管理时拥有全部权限，但是当顾客身份登录时，便会有权限不够的时候，如图 20、21、22 所示。并且同样是操作 3，顾客在找不到时，无法进行添加的操作，如图 23 所示。

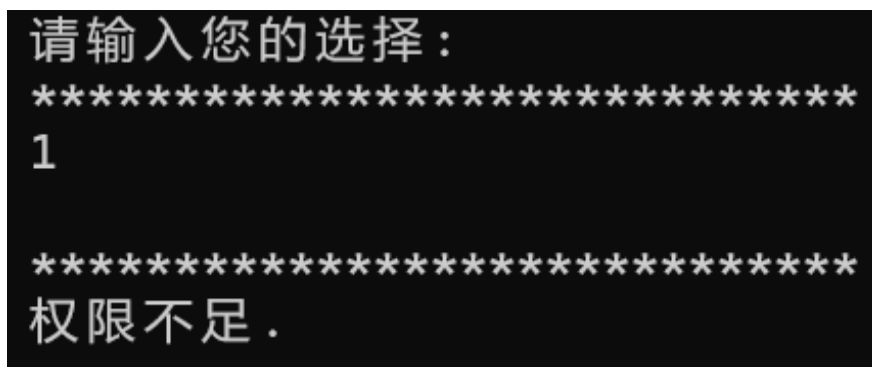


图 20 权限不够图



图 21 权限不够图





图 22 权限不够图

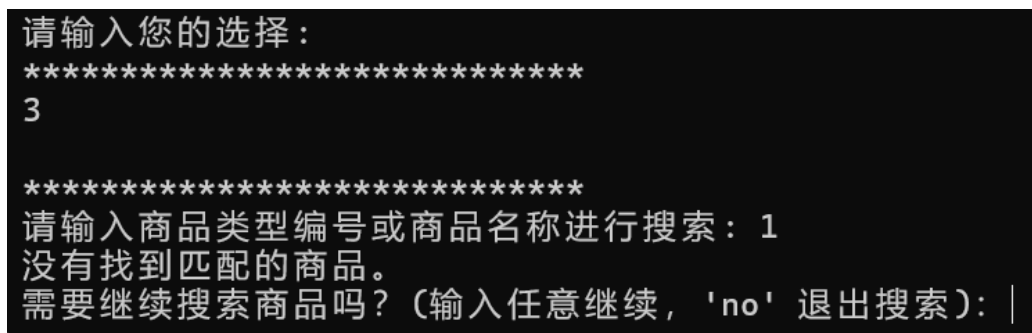


图 23 顾客针对 3 的权限图

## 课程设计总结

### 设计特点：

首先本课程设计是一个为了实现商品销售管理的系统。商品信息括商品编号、商品名称、进货价格、销售价格、商品库存等。该系统系统采用菜单工作方式，保证用户界面清晰、友好、美观、易用。商品信息导入功能，可以从磁盘文件导入商品信息，支持 csv。提供商品信息的查看功能，可以输出所有商品的详细信息。提供商品查询功能，允许按类型或商品名称进行搜索，并将结果显示出来。新增商品信息功能，可以向商品库中添加新的商品信息。进货功能，进货时填写商品编号和进货数量，商品库存随之增加。销售功能，顾客购买商品时，输入商品编号，系统会自动生成销售清单，并统计本次销售的总额，同时商品库存会有所减少。

### 创新之处：

- 1、本系统创新的添加了用户身份管理系统和数据保存功能，身份的限制能够更好的保护系统安全。
- 2、本系统支持数据导出导入，同时导入数据时会进行合法性筛查，保证 id 和名称不重复，并且存储到表格的数据也是合法的。而且将数据的导出方便了下一次的数据导入。

### 不足之处：

- 1、本系统并没有制作新增商品属性的功能，商品的属性只局限于商品编号、商品名称、进货价格、销售价格、商品库存等。
- 2、为了方便在输入新商品的名字时限定死了是英文。

过程中遇到的困难及解决方法：

1、设计中遇到的问题也是最后调试完的是由于身份的限制，在操作 3 时，因为原本将添加函数写在搜索函数内部，导致一开始，没有思绪来控制操作三的权限，使得经手是顾客身份也能对所有数据进行操作。解决办法是，通过查询资料，发现可以将 role 作为参数传给 `searchProduct(role)`，然后在其中通过增加 `if(role==ROLE_ADMIN)` 的判断，使得添加操作只有管理员才能进行。

2、遇到的第二个困难是无法实现登录失败后再次重新登录，简单的 `break` 无法直接跳到我想要的位置，导致无法重复登录，解决办法是增加了一个大的 `while` 循环条件在外部，当跳转到 `else()` 登录错误，中的 `case2` 时再次进行用户登录 `role = userAuthentication()`；重新赋值 role 使得在下一轮循环时可以直接进入正确环节。

3、也是一个小问题是字符编码的问题，中文乱码，查询资料后添加了一行 `SetConsoleOutputCP(CP_UTF8);` //中文显示便解决了。

心得、感想：通过这次课设练习，对结构体的理解和对身份设置的理解都更进一步，对循环条件的嵌套和退出，都有了新的体会。明白了在 c 语言当中如何通过指针和列表来实现以往在 python 中对数组的操作。同时对编译器报错的理解也更加的熟悉，能够分辨出每一个问题大致的意思，并根据 vscode 的建议进行修改。是对编程能力的一次极大锻炼