

东软秘密

**设计报告  
*（项目名称）***

东软集团股份有限公司IT人才实训中心

版权所有，翻版必究

**更改履历**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 版本 | 更改时间 | 更改人 | 更改章节 | 状态 | 更改描述 |
| 1 | V1.0 | 2019-7-25 | 孙超· |  | 新建 |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

状态：新建、增加、修改、删除。

目 录

[1 引言 4](#_Toc15045123)

[1.1 编制目的 4](#_Toc15045124)

[1.2 词汇表 4](#_Toc15045125)

[1.3 参考资料 4](#_Toc15045126)

[2 系统开发环境 5](#_Toc15045127)

[3 系统设计思路 5](#_Toc15045128)

[4 系统公用代码设计 5](#_Toc15045129)

[4.1 显示菜单类： 5](#_Toc15045130)

[4.2 管理员操作类： 9](#_Toc15045131)

[4.3 消费者购买类 18](#_Toc15045132)

[4.4 异常处理 24](#_Toc15045133)

[5 功能模块设计 24](#_Toc15045134)

[*5.1* *登陆界面* 24](#_Toc15045135)

[*5.2* *过期检测* 26](#_Toc15045136)

[6 数据库设计 27](#_Toc15045138)

[6.1 功能说明 27](#_Toc15045139)

[6.2 数据表设计 28](#_Toc15045140)

[6.3 视图设计 28](#_Toc15045141)

[6.4 序列设计 28](#_Toc15045142)

# 引言

## 编制目的

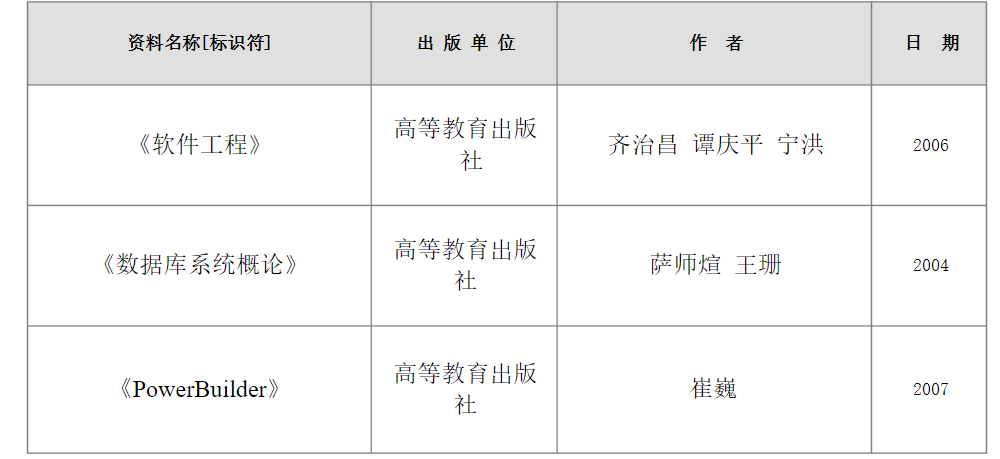
本报告详细完成对*自动售货*系统的整体设计，达到指导开发的目的，同时实现和测试人员及用户的沟通。

本报告面向详细设计人员、开发人员、测试人员及最终用户而编写，是了解系统的导航。

## 词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| 商品 | 售货机里的商品 | goods |
| 销售商品 | 已卖的商品 | sold |
| 货币 | 售货机里的现金货币 | money |
|  |  |  |
|  |  |  |

## 参考资料



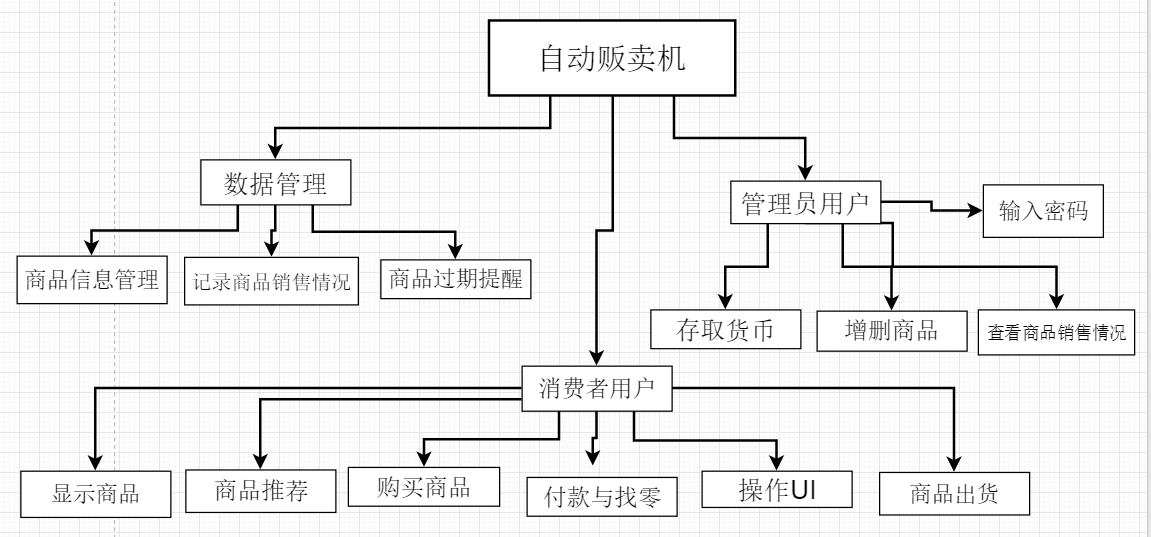
# 系统开发环境

操作系统： *Windows 10*

集成开发工具：*VS2017*

编译环境：

# 系统设计思路



# 系统公用代码设计

## 显示菜单类：

### 功能描述

*用来显示售货机的操作菜单，显示商品属性的菜单，管理员管理商品的菜单，用户选择付款方式时的选择菜单，管理员的操作菜单等*

### 类、方法描述

程序从文件中读取数据，并通过printf方法将数据按照一定的格式输出在控制台上。

#### *函数具体信息*

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 | 方法名 | 功能 | 参数说明 |
| **void** | **printGoodsMenu** | **用来显示给消费者看的商品信息菜单** | **goods \*good, int i** |
| **int** | **menu** | **用来显示选择菜单，用户可选择消费者模式和管理员模式，返回用户的选择** |  |
| **int** | **adminmenu** | **用来显示管理员操作界面的菜单，有多种选项供管理员选择，函数返回管理员选择的选项** |  |
| **void** | **printallgoodsmenu** | **用来显示管理员管理商品信息的菜单，显示商品所有属性（包括消费者无法看到的）** | **goods \*good, int i** |

void printGoodsMenu(goods \*good, int i) {

int j;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("编号 商品 单价\n");

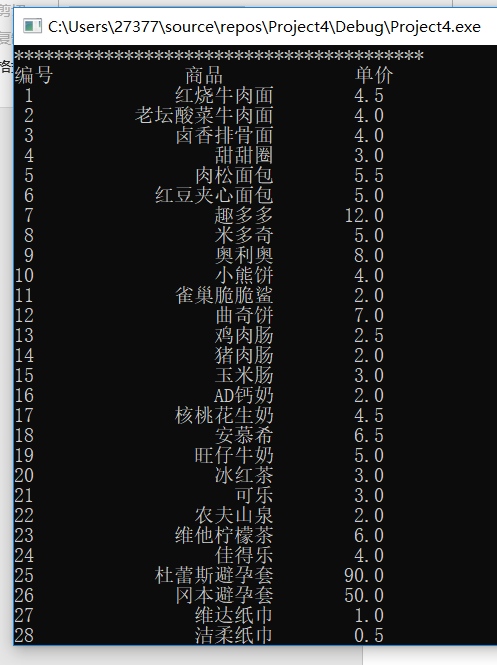
for (j = 0; j < i; j++) {

printf("%2d \t %16s \t %4.1f\n", good[j].num, good[j].name, good[j].price);

}

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n");

}//打印商品菜单



int menu() {

int a;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("输入1来显示商品菜单\n输入2来进入管理员系统\n输入0来退出程序\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n");

scanf("%d", &a);

while (1) {

if (a == 1 || a == 2 || a == 0) {

return a;

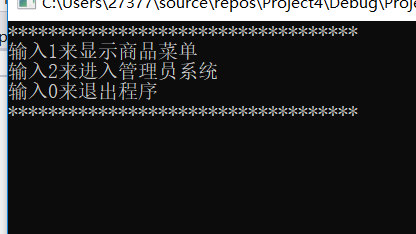
}

printf("wrong choice!!!\n");

scanf("%d", &a);

}

}//打印操作界面



int adminmenu() {

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n —— 管理员菜单 —— \n");

printf("输入1显示商品全部信息\n输入2查看需要补货的商品 \n输入3上架新种类商品\n输入4下架商品\n输入5管理售货机现金\n输入6查看过期商品\n输入7查看销售情况\n输入0退出系统\n");

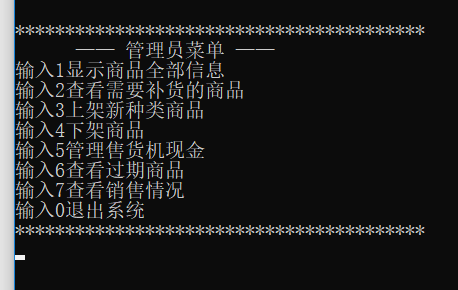
printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

int choice;

scanf("%d", &choice);

return choice;

}//显示管理员操作菜单



void printallgoodsmenu(goods \*good, int i) {

int j;

printf("编号 数量 已卖数 名字 价格 到期日期 \n");

for (j = 0; j < i; j++)

printf( "%2d %2d %2d %16s\t %4.1f %4d %2d %2d \n", good[j].num, good[j].amount, good[j].sold, good[j].name, good[j].price, good[j].date[0], good[j].date[1], good[j].date[2]);

}//打印商品的所有详细属性，包括编号，货余量，名字，已卖数，价格，到期日期等



## 管理员操作类：

### 功能描述

*管理员查看商品属性，增加减少商品种类，补货，管理售货机货币，查看销量和过期商品等操作。*

### 类，方法描述

管理员通过控制台输入相关命令，可以更改该程序的相关文件中的数据，从而管理售货机系统。

#### *函数具体信息*

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 | 方法名 | 功能 | 参数说明 |
| void | addgoods | 管理员对售货机内的商品进行补货 | goods \*good, int i |
| void | addnewkindgoods | 管理员增加新商品 | goods \*good, int \*i |
| void | ItemDownshelf | 管理员下架商品 | goods \*good, int \*i |
| void | managedmoney | 从money.txt中读取售货机内的货币数，管理员可以取出或存放货币 |  |
| void | checkoverdue | 管理员查看商品过期情况，过期商品无法被购买 | goods \*good, int i |
| void | checksales | 管理员查看销售情况，系统会自动统计销量并提示哪些商品的销量不佳，并建议更换 | goods \*good, int i |
| void | Administrator | 管理员登陆界面，实现文件读写 | goods \*good, int \*i |

void Administrator(goods \*good, int \*i) {

int key,j;

FILE \*fpgoods;

printf("请输入管理员密码（密码是1234）\n");

scanf("%d", &key);

if (key == 1234) {

while (1)

{

int choice = adminmenu();

switch (choice)

{

case 1:printallgoodsmenu(good,\*i); break;

case 2:addgoods(good, \*i); break;

case 3:addnewkindgoods(good, i); break;

case 4:ItemDownshelf(good,i); break;

case 5:managedmoney(); break;

case 6:checkoverdue(good,\*i); break;

case 7:checksales(good,\*i); break;

case 0:break;

default:

break;

}

if (choice == 0)

break;

if ((fpgoods = fopen("goods.txt", "w")) == NULL) {

printf("文件打开失败！！！\n");

return;

}

for (j = 0; j < \*i; j++)

fprintf(fpgoods, "%d %d %d %s %f %d %d %d %d\n", good[j].kind, good[j].amount, good[j].sold, good[j].name, good[j].price, good[j].date[0], good[j].date[1], good[j].date[2], good[j].num);

fclose(fpgoods);

}

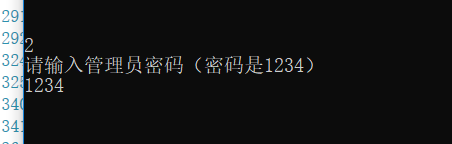
}

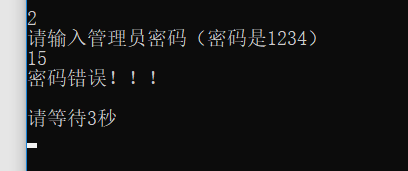
else {

printf("密码错误！！！\n\n");

}

}





void addgoods(goods \*good, int i) {

int j;

int flag = 0;

int num;

int amount;

for (j = 0; j < i; j++) {

if (good[j].amount <=2) {

flag++;

if (flag == 1)

printf("需要补货的商品有：\n\n");

printf("%3d %8s\n", good[j].num, good[j].name);

}

}

if (flag == 0) {

printf("没有需要补货的商品!!!\n\n\n");

return;

}

printf("请输入补货商品的编号：(输入0退出补货）\n");

scanf("%d", &num);

if (num == 0)

return;

if (num > 0 && num <= i) {

for (j = 0; j < i; j++) {

if (good[j].num == num) {

good[j].amount = 10;

printf("补货成功！！\n");

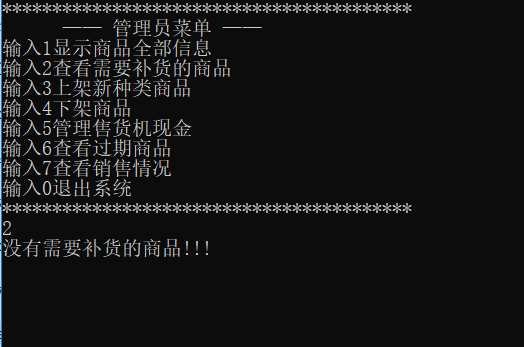
printf("编号：%3d 名称：%8s 数量：%d\n", good[j].num, good[j].name, good[j].amount);

}

}

}

}//补货操作，当商品的货物少于等于2时，系统会提醒管理员进行补货，且管理员进行一次补货操作时会直接将货补充至10个（自动贩卖机每一种货物的上限就是10，位置有限）



void addnewkindgoods(goods \*good, int \*i) {

good[\*i].num = (\*i)+1;

good[\*i].sold = 0;

good[\*i].amount = 10;

printf("请输入新商品名称\n");

scanf("%s", good[\*i].name);

printf("请输入新商品价格\n");

scanf("%f", &good[\*i].price);

printf("请输入到期年月日（以空格隔开）\n");

scanf("%d %d %d", &good[\*i].date[0], &good[\*i].date[1], &good[\*i].date[2]);

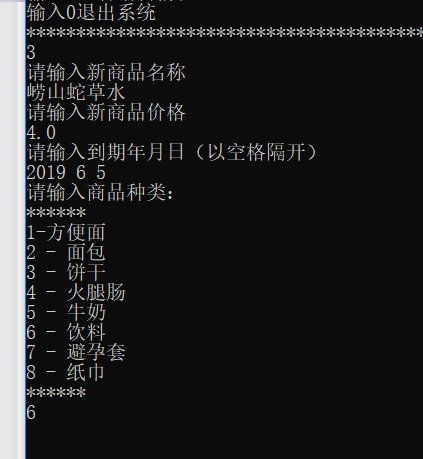
printf("请输入商品种类：\n");

printf("\*\*\*\*\*\*\n1-方便面\n2 - 面包\n3 - 饼干\n4 - 火腿肠\n5 - 牛奶\n6 - 饮料\n7 - 避孕套\n8 - 纸巾\n\*\*\*\*\*\*\n");

scanf("%d", &good[\*i].kind);

(\*i)++;

}



void ItemDownshelf(goods \*good, int \*i) {

printallgoodsmenu(good, \*i);

int num, j;

while (1) {

printf("请输入下架商品的正确编号（按0退出）\n");

scanf("%d", &num);

if (num > \*i || num < 0)

continue;

if (num > 0)

for (j = 0; j < \*i; j++) {

if (good[j].num == num)

{

for (; j < (\*i) - 1; j++) {

good[j] = good[j + 1];

good[j].num = j+1;

}

}

}

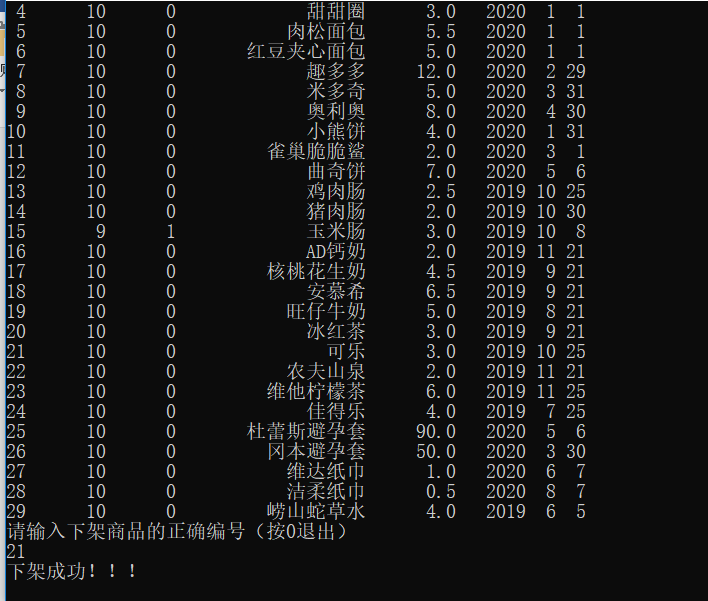
(\*i)--;

printf("下架成功！！！\n\n");

break;

}

}//按编号下架某种商品



void managedmoney() {

FILE \*fpmoney;

float money;

int choice;

float change;

if ((fpmoney = fopen("money.txt", "r")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fscanf(fpmoney, "%f", &money);

fclose(fpmoney);

printf("当前售货机内还有余额：%.1f", money);

while (1) {

printf("\n输入1取出余额 输入2增加货币 输入0退出\n");

scanf("%d", &choice);

if (choice == 1) {

while (1) {

printf("请输入取出的数目：(输入0退出)\n");

scanf("%f", &change);

if ((money - change) <= 500) {

printf("售货机内必须存有至少500元现金！！！\n\n");

continue;

}

if (change == 0) {

break;

}

if ((money - change) > 500) {

money -= change;

printf("取款成功！！\n");

printf("当前售货机余额： %.1f", money);

if ((fpmoney = fopen("money.txt", "w")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fprintf(fpmoney, "%f", money);

fclose(fpmoney);

}

}

}

else if (choice == 2) {

printf("请输入增加的数目：\n");

scanf("%f", &change);

money += change;

printf("货币储存成功！！！\n");

printf("当前售货机余额： %.1f", money);

if ((fpmoney = fopen("money.txt", "w")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fprintf(fpmoney, "%f", money);

fclose(fpmoney);

}

else if (choice == 0)

break;

else {

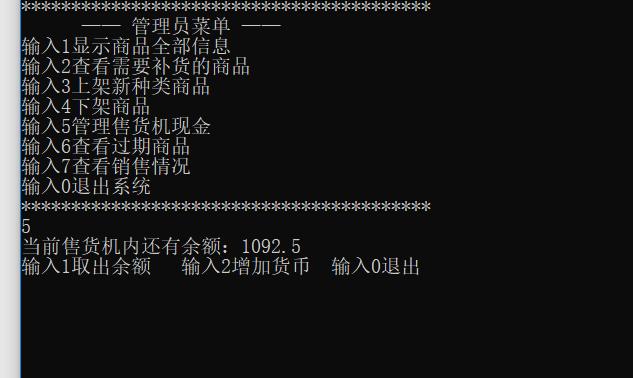
printf("请输入正确的选项！！\n");

continue;

}

}

}



void checkoverdue(goods \*good, int i) {

int j;

time\_t t;

struct tm \* lt;

time(&t);//获取Unix时间戳。

lt = localtime(&t);//转为时间结构。

//printf("%d/%d/%d %d:%d:%d\n", lt->tm\_year + 1900, lt->tm\_mon, lt->tm\_mday, lt->tm\_hour, lt->tm\_min, lt->tm\_sec);//输出结果

printf("过期商品有：\n");

printf("编号 \t \t 名称\t\t \t 到期日期\t\n");

for (j = 0; j < i; j++) {

if (good[j].date[0] > lt->tm\_year + 1900) {

continue;

}

if (good[j].date[0] == lt->tm\_year + 1900&&good[j].date[1]> lt->tm\_mon) {

continue;

}

if (good[j].date[0] == lt->tm\_year + 1900 && good[j].date[1] == lt->tm\_mon + 1 && good[j].date[2] > lt->tm\_mday) {

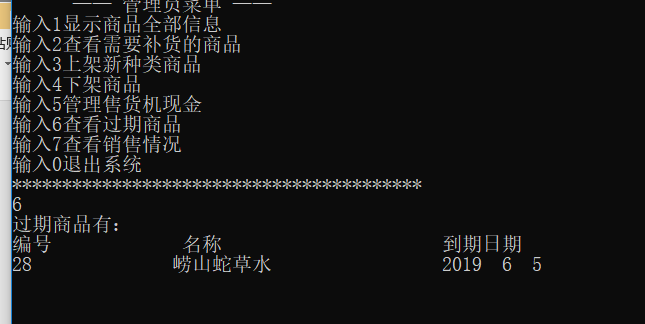
continue;

}

printf("%2d \t \t%8s\t\t %4d %2d %2d\n", good[j].num, good[j].name, good[j].date[0], good[j].date[1], good[j].date[2]);

}

}



void checksales(goods \*good, int i) {

FILE \*fpsold;

sold \* head;

sold \*p, \*s,\*q;

time\_t t;

struct tm \* lt;

time(&t);//获取Unix时间戳。

lt = localtime(&t);//转为时间结构。

//printf("%d/%d/%d %d:%d:%d\n", lt->tm\_year + 1900, lt->tm\_mon, lt->tm\_mday, lt->tm\_hour, lt->tm\_min, lt->tm\_sec);//输出结果

int num = 0;

int j;

head = (sold \*)malloc(sizeof(sold));

if (head == NULL) exit(1);

head->next = NULL;

s = head;

if ((fpsold = fopen("sold.txt", "r"))==NULL) {

printf("销售文件打开失败！！\n");

return;

}

while (!feof(fpsold)) {

p = (sold \*)malloc(sizeof(sold));

fscanf(fpsold, "%d %s %f %d %d %d %d %d %d\n", &p->num, p->name, &p->price, &p->sold\_year, &p->sold\_month, &p->sold\_day, &p->sold\_hour,&p->sold\_min, &p->sold\_sec);

s->next = p;

s = p;

}

s->next = NULL;

q = head->next;

while (q != NULL) {

printf("%d %s %.1f 销售时间：%d-%d-%d-%d时%d分%d秒\n", q->num, q->name, q->price,q->sold\_year,q->sold\_month,q->sold\_day,q->sold\_hour,q->sold\_min,q->sold\_sec);

q = q->next;

}

for (j = 0; j < i; j++) {

num += good[j].sold;

}

if (num > 100) {

printf("\n\n以下商品销量不佳，若不是刚上架的新品，建议更换\n\n");

printf("编号 \t \t名称\t\t已卖数\t\n");

for (j = 0; j < i; j++) {

if (good[j].sold <= (num / i) / 5)

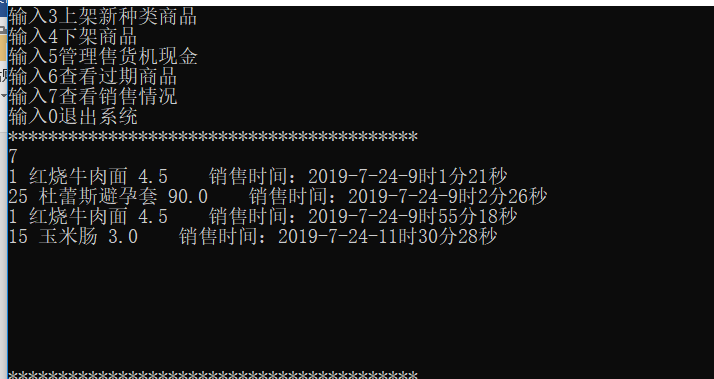
printf("%2d\t%16s\t%d\n", good[j].num, good[j].name, good[j].sold);

}

}

printf("\n\n\n");

}



## 消费者购买类

### 功能说明

*消费者通过该程序可以购买自动贩卖机里的商品*

### 类，方法描述

消费者通过控制台输入相关命令，可以购买商品，系统会根据消费者的行为来修改文件中的数据。

#### *函数具体信息*

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 | 方法名 | 功能 | 参数说明 |
| **void** | **buyGoods** | **消费者购买商品，且系统会自动检测过期商品** | **goods \*good, int i** |
| **void** | **cashpay** | **付款时现金支付，系统会改写文件中的数据** | **float price, int j** |

void cashpay(float price, int j) {

int pay;

FILE \*fpsold;

FILE \*fpmoney;

float money;

time\_t t;

struct tm \* lt;

time(&t);//获取Unix时间戳。

lt = localtime(&t);//转为时间结构。

if ((fpmoney = fopen("money.txt", "r")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fscanf(fpmoney, "%f", &money);

fclose(fpmoney);

while (1) {

printf("请输入您投入的现金数\n\n");

scanf("%d", &pay);

//现金不足以完成支付时，用户可以换成扫码支付或者退出交易

if (pay < price) {

int choice;

printf("您输入的现金无法完成购买！！\n");

printf("输入1继续购买 输入2退出此次购买 输入3使用支付宝购买 输入4使用微信购买\n ");

scanf("%d", &choice);

if (choice == 1) {

continue;

}

else if (choice == 2) {

printf("购买失败！！\n\n\n\n\n");

break;

}

else if (choice == 3 || choice == 4) {

printf("请扫码完成支付！！\n\n\n");

printf("\t\t\t\*\*\* \*\* \*\*\* \*\*\* \n");

printf("\t\t\t\* \*\* \*\*\*\*\*\*\* \*\* \n");

printf("\t\t\t\* \*\*\*\* \*\*\*\*\* \* \n");

printf("\t\t\t\*\* \*\*\*\*\* \* \*\* \n");

printf("\t\t\t\* \*\* \*\* \*\* \n");

printf("\t\t\t\*\*\* \* \*\* \*\*\*\*\*\* \n");

printf("\t\t\t\* \*\*\*\* \*\* \*\* \*\* \n");

printf("\t\t\t\* \*\* \*\* \*\* \*\*\*\*\* \n");

printf("\t\t\t\* \*\* \*\* \*\*\*\*\*\*\*\* \n\n\n\n");

printf("交易完成，请与出货口取出商品\n\n");

good[j].amount--;

good[j].sold++;

if ((fpsold = fopen("sold.txt", "a")) == NULL) {

printf("售卖记录文件打开失败！！\n\n");

return;

}

fprintf(fpsold, "%d %s %.1f %d %d %d %d %d %d\n", good[j].num, good[j].name, good[j].price, lt->tm\_year + 1900, lt->tm\_mon + 1, lt->tm\_mday, lt->tm\_hour, lt->tm\_min, lt->tm\_sec);

fclose(fpsold);

break;

}

}

//现金充足时，提供找零服务

else if(pay >= price&&money>=(pay-price)){

if (pay == price) {

printf("交易完成，请与出货口取出商品\n\n");

good[j].amount--;

good[j].sold++;

money += price;

if ((fpmoney = fopen("money.txt", "w")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fprintf(fpmoney, "%f", money);

fclose(fpmoney);

if ((fpsold = fopen("sold.txt", "a")) == NULL) {

printf("售卖记录文件打开失败！！\n\n");

return;

}

fprintf(fpsold, "%d %s %.1f %d %d %d %d %d %d\n", good[j].num, good[j].name, good[j].price, lt->tm\_year + 1900, lt->tm\_mon + 1, lt->tm\_mday, lt->tm\_hour, lt->tm\_min, lt->tm\_sec);

fclose(fpsold);

break;

}

else {

printf("找零%.1f元，请与零钱出口取出找零\n", pay - price);

printf("交易完成，请与出货口取出商品\n\n\n\n\n");

good[j].amount--;

good[j].sold++;

money += price;

if ((fpmoney = fopen("money.txt", "w")) == NULL) {

printf("货币文件打开失败！！\n\n");

return;

}

fprintf(fpmoney, "%f", money);

fclose(fpmoney);

if ((fpsold = fopen("sold.txt", "a")) == NULL) {

printf("售卖记录文件打开失败！！\n\n");

return;

}

fprintf(fpsold, "%d %s %.1f %d %d %d %d %d %d\n", good[j].num, good[j].name, good[j].price, lt->tm\_year + 1900, lt->tm\_mon + 1, lt->tm\_mday, lt->tm\_hour, lt->tm\_min, lt->tm\_sec);

fclose(fpsold);

break;

}

}

else {

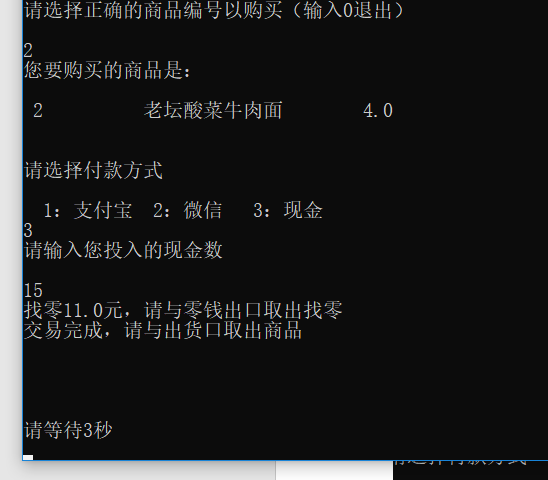
printf("\n售货机零钱不够，无法找零，请换其他支付方式或者面额更小的纸币\n\n");

break;

}

}

}//现金支付流程



void buyGoods(goods \*good, int i) {

int choice;

int j, k;

int way;

FILE \*fpsold;

time\_t t;

struct tm \* lt;

time(&t);//获取Unix时间戳。

lt = localtime(&t);//转为时间结构。

while (1) {

printf("请选择正确的商品编号以购买（输入0退出）\n\n");

scanf("%d", &choice);

if (choice == 0)

return;

if (choice<0 || choice>i)

{

printf("wrong choice!!\n\n");

continue;

}

break;

}

for (j = 0; j < i; j++) {

if (good[j].num == choice) {

if (good[j].date[0] < lt->tm\_year + 1900) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].date[0] == lt->tm\_year + 1900 && good[j].date[1] < lt->tm\_mon) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].date[0] == lt->tm\_year + 1900 && good[j].date[1] == lt->tm\_mon + 1 && good[j].date[2] < lt->tm\_mday) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].amount > 0) {

printf("您要购买的商品是：\n\n");

printf("%2d \t %16s \t %4.1f\n\n\n", good[j].num, good[j].name, good[j].price);

printf("请选择付款方式\n\n 1：支付宝 2：微信 3：现金\n");

break;

}

else {

printf("该商品无货！！！\n\n\n");

//智能推荐

printf("以下商品与您选择的商品类似，您可以考虑购买以下商品\n\n\n");

for (k = 0; k < i; k++) {

if (good[k].kind == good[j].kind&&good[k].amount > 0) {

printf("%2d \t %16s \t %4.1f\n", good[k].num, good[k].name, good[k].price);

}

}

printf("\n\n");

return;

}

}

}

scanf("%d", &way);

while (1) {

if (way == 1 || way == 2) {

printf("请扫码完成支付！！\n\n\n");

printf("\t\t\t\*\*\* \*\* \*\*\* \*\*\* \n");

printf("\t\t\t\* \*\* \*\*\*\*\*\*\* \*\* \n");

printf("\t\t\t\* \*\*\*\* \*\*\*\*\* \* \n");

printf("\t\t\t\*\* \*\*\*\*\* \* \*\* \n");

printf("\t\t\t\* \*\* \*\* \*\* \n");

printf("\t\t\t\*\*\* \* \*\* \*\*\*\*\*\* \n");

printf("\t\t\t\* \*\*\*\* \*\* \*\* \*\* \n");

printf("\t\t\t\* \*\* \*\* \*\* \*\*\*\*\* \n");

printf("\t\t\t\* \*\* \*\* \*\*\*\*\*\*\*\* \n\n\n\n");

printf("交易完成，请与出货口取出商品\n\n");

good[j].amount--;

good[j].sold++;

if ((fpsold = fopen("sold.txt", "a")) == NULL) {

printf("售卖记录文件打开失败！！\n\n");

return;

}

fprintf(fpsold, "%d %s %.1f %d %d %d %d %d %d\n", good[j].num, good[j].name, good[j].price, lt->tm\_year + 1900,lt->tm\_mon+1, lt->tm\_mday,lt->tm\_hour,lt->tm\_min,lt->tm\_sec);

fclose(fpsold);

break;

}

else if (way == 3) {

cashpay(good[j].price, j);

break;

}

else {

printf("wrong choice!!!!!\n");

printf("请重新选择付款方式！(或者按0退出交易）\n");

scanf("%d", &way);

if (way == 0)

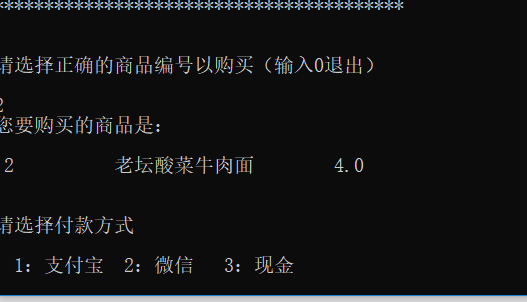
break;

else

continue;

}

}



## 异常处理

### 功能说明

*对用户的输入进行了防错处理，比如需要用户输入0或1，而用户却输入了2，那么系统就会提示用户输入错误并让用户继续输入，直到正确为止（用户也可选择退出）。*

### 设计说明

*用while循环加if判断来处理输入错误。*

# 功能模块设计

*从本章开始，逐个地给出各个功能模块的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。*

## *登陆界面*

### 功能说明

*登陆管理员系统输入密码的界面*

### 类、方法设计

#### 函数具体信息

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 | 方法名 | 功能 | 参数说明 |
| void | Administrator | 系统判断密码是否正确来决定是否能进入管理员操作系统 | goods \*good, int \*i |

void Administrator(goods \*good, int \*i) {

int key,j;

FILE \*fpgoods;

printf("请输入管理员密码（密码是1234）\n");

scanf("%d", &key);

if (key == 1234) {

while (1)

{

int choice = adminmenu();

switch (choice)

{

case 1:printallgoodsmenu(good,\*i); break;

case 2:addgoods(good, \*i); break;

case 3:addnewkindgoods(good, i); break;

case 4:ItemDownshelf(good,i); break;

case 5:managedmoney(); break;

case 6:checkoverdue(good,\*i); break;

case 7:checksales(good,\*i); break;

case 0:break;

default:

break;

}

if (choice == 0)

break;

if ((fpgoods = fopen("goods.txt", "w")) == NULL) {

printf("文件打开失败！！！\n");

return;

}

for (j = 0; j < \*i; j++)

fprintf(fpgoods, "%d %d %d %s %f %d %d %d %d\n", good[j].kind, good[j].amount, good[j].sold, good[j].name, good[j].price, good[j].date[0], good[j].date[1], good[j].date[2], good[j].num);

fclose(fpgoods);

}

}

else {

printf("密码错误！！！\n\n");

}

### }

### 相关数据表

*无*

## *过期检测*

### 功能说明

用户购买商品时系统会自动检测该商品是否过期

### 类、方法设计

#### 函数具体信息

|  |  |  |  |
| --- | --- | --- | --- |
| 返回值 | 方法名 | 功能 | 参数说明 |
| void | buygoods | 用户购买商品,且系统会自动检测该商品是否过期 | goods \*good, int \*i |

FILE \*fpsold;

time\_t t;

struct tm \* lt;

time(&t);//获取Unix时间戳。

lt = localtime(&t);//转为时间结构。

while (1) {

printf("请选择正确的商品编号以购买（输入0退出）\n\n");

scanf("%d", &choice);

if (choice == 0)

return;

if (choice<0 || choice>i)

{

printf("wrong choice!!\n\n");

continue;

}

break;

}

for (j = 0; j < i; j++) {

if (good[j].num == choice) {

if (good[j].date[0] < lt->tm\_year + 1900) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].date[0] == lt->tm\_year + 1900 && good[j].date[1] < lt->tm\_mon) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].date[0] == lt->tm\_year + 1900 && good[j].date[1] == lt->tm\_mon + 1 && good[j].date[2] < lt->tm\_mday) {

printf("该商品已过期，无法购买！！！\n\n");

return;

}

if (good[j].amount > 0) {

printf("您要购买的商品是：\n\n");

printf("%2d \t %16s \t %4.1f\n\n\n", good[j].num, good[j].name, good[j].price);

printf("请选择付款方式\n\n 1：支付宝 2：微信 3：现金\n");

break;

}

else {

printf("该商品无货！！！\n\n\n");

//智能推荐

printf("以下商品与您选择的商品类似，您可以考虑购买以下商品\n\n\n");

for (k = 0; k < i; k++) {

if (good[k].kind == good[j].kind&&good[k].amount > 0) {

printf("%2d \t %16s \t %4.1f\n", good[k].num, good[k].name, good[k].price);

}

}

printf("\n\n");

return;

}

}

# }

# 数据库设计

## 功能说明

商品数据存储在good.txt中，售货机货币数据储存在money.txt中。且要修改这些文件中的数据需要管理员密码。

销售记录存储在sold.txt中，且只能由程序运行时自己写入数据，管理员和用户都无法修改该文件中的数据。

## 数据表设计

### *表名：goods 表功能说明*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **可为空** | **默认** | **注释** |
| num | Int | 否 | 无 | 商品编号 |
| amount | Int | 否 | 10 | 商品数量 |
| sold | Int | 否 | 0 | 商品已卖数 |
| name | Char[20] | 否 | 无 | 商品名称 |
| price | Float | 否 | 无 | 商品价格 |
| Data0 | Int | 否 | 无 | 商品到期日期（年） |
| Data1 | Int | 否 | 无 | 商品到期日期（月） |
| Data2 | int | 否 | 无 | 商品到期日期（日） |

### *表名：sold 表功能说明*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **可为空** | **默认** | **注释** |
| num | Int | 否 | 无 | 商品编号 |
| name | Char[20] | 否 | 无 | 商品名称 |
| price | Float | 否 | 无 | 商品价格 |
| Data0 | Int | 否 | 无 | 商品售出日期（年） |
| Data1 | Int | 否 | 无 | 商品售出日期（月） |
| Data2 | int | 否 | 无 | 商品售出日期（日） |
| Data3 | int | 否 | 无 | 商品售出日期（时） |
| Data4 | Int | 否 | 无 | 商品售出日期（分） |
| Data5 | int | 否 | 无 | 商品售出日期（秒） |
|  |  |  |  |  |

### *表名：money 表功能说明*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **可为空** | **默认** | **注释** |
| money | float | 否 | 1000 | 售货机存储货币数 |

## 视图设计

## 序列设计