**华南农业大学数学与信息学院**

**课程论文（设计）**

2015-2016　学年第2学期

课程名称：数据结构课程设计

课程性质：实践课

专业班级：2014级信息与计算科学3 班

考核方式：考查

学生姓名：何唯

学生学号：2014330120308

学 　分：1

学 时：1周

教师姓名：廖彬

|  |
| --- |
| **自评分：100**  **评语及评分** |

**目 录**

[1. 需求分析 4](#_Toc15526)

[1.1 问题分析 4](#_Toc29011)

[1.2 用户需求分析 5](#_Toc418)

[1.3 功能需求分析 5](#_Toc32352)

[1.3.1 功能模块说明： 5](#_Toc5393)

[1.3.2 订票系统功能图 6](#_Toc10353)

[1.3.3 系统需求分析 6](#_Toc11624)

[2. 概要设计 7](#_Toc20717)

[2.1 数据结构设计说明 7](#_Toc16407)

[2.1.1 航班信息 7](#_Toc21148)

[2.1.2 客户订单信息 8](#_Toc29554)

[2.2 各功能模块调用 9](#_Toc24173)

[2.3 函数说明 10](#_Toc28914)

[3. 详细设计 11](#_Toc19252)

[3.1 添加航班信息 11](#_Toc8813)

[3.2 浏览航班以及订票信息 11](#_Toc22643)

[3.3 查找航班信息模块 12](#_Toc31293)

[3.4 客户订票模块 12](#_Toc9336)

[3.5 退票模块 13](#_Toc9581)

[3.6 修改航班信息模块 14](#_Toc22607)

[3.7 航班信息取消模块 15](#_Toc16474)

[4. 调试分析 15](#_Toc24663)

[5. 总结 23](#_Toc10615)

[6. 附录 23](#_Toc32386)

1. 需求分析

1.1 问题分析

设计飞机订票系统：通过此系统可以实现如下功能：

* 录入：可以录入航班情况（数据可以存储在一个数据文件中，数据结构、具体数据自定）
* 查询：可以查询某个航线的情况（如，输入航班号，查询起降时间，起飞抵达城市，航班票价，票价折扣，确定航班是否满仓）；可以输入起飞抵达城市，查询飞机航班情况；
* 订票：（订票情况可以存在一个数据文件中，结构自己设定）可以订票，如果该航班已经无票，可以提供相关可选择航班；
* 退票： 可退票，退票后修改相关数据文件；客户资料有姓名，证件号，订票数量及航班情况，订单要有编号。
* 修改航班信息：当航班信息改变可以修改航班数据文件

——根据以上功能说明，设计航班信息，订票信息的存储结构，设计程序完成功能；

1.2 用户需求分析

用户使用此程序所要完成的工作主要为：录入和查询所有航线信息、增加航线信息、查看已订票订单信息、查询航线、办理订票业务、办理退票业务、航班取消更新服务。通过此系统可以方便的进行上述工作。

每条航线所涉及的信息有：航班号、起始站、终点站、起飞时间、降落时间、机票数、机票价格。

已订票的订单客户名单信息包括：订单号、客户身份ID、客户名字、航班号、所定票数、订单价钱。

查询航线功能可以根据旅客提出的起终点站名输出航班号、起始站、终点站、起飞时间、降落时间、机票数、机票价格。（包括相同航线所有航班信息）

订票业务功能根据客户提出的要求（航班号、订票数额）查询该航班票额情况，若尚有余票，则为客户办理订票手续；若余票额少于定票额，则需重新询问客户要求。若已满员，则推荐提供其余相同航线有余位的航班。

退票业务功能根据客户提供的情况（订单号）为客户办理退票手续。若退票数大于已定票数，则输出错误，重新询问客户要求。若退票数满足条件，则为他办理订票手续。

航班取消更新服务根据航班突发情况，输入航班号，航班公司可取消该航班，并更新至数据文件，取消相关订单。

1.3 功能需求分析

1.3.1 功能模块说明：

(1) 将航班的基本信息录入到系统中，与数据文件同步。

(2) 根据起飞地点、降落地点查询所需航班信息。

(3) 输出所有航班信息以及订单信息。

(4) 输入姓名和身份证号码以及所要订航班的起降地点订票，输出订单号码以及费用总额，即生成订单信息。

(5) 输入用户订单号办理退票，输出退款总额。

(6) 输入航班号，可修改该航班的起飞和降落时间、修改航班的起降地点、票价、剩余票数以及航班号。

(7) 退出或者模块功能使用结束的同时将信息写入数据文件，进入程序读入数据文件。

1.3.2 订票系统功能图

**飞机票订票系统**

**订票业务**

**查找航班信息**

**退出系统**

**航班取消**

**航班及订票信息**

**x退票业务**

**修改航班信息**

**增加航班信息**

图1.订票系统结构图

1.3.3 系统需求分析

开发环境：PC 机，Windows 8.1 系统

运行软件：CodeBlocks V13.12

2. 概要设计

2.1 数据结构设计说明

2.1.1 航班信息

航班的情况存储结构采用顺序表，每个元素表示一个航班的情况，包括航班号、起始站、终点站、起飞时间、降落时间、剩余机票数、机票价格七个数据项。如下表所示：

|  |  |  |
| --- | --- | --- |
| 数据项 | 数据类型 | 数据说明 |
| num | 整型(int) | 航班号 |
| start | 字符串(char[30]) | 起始站 |
| destination | 字符串(char[30]) | 终点站 |
| TakeOffTime | 字符串(char[30]) | 起飞时间 |
| ArriveTime | 字符串(char[30]) | 降落时间 |
| count | 整型(int) | 剩余机票数 |
| price | 整型(int) | 机票价格 |

表1 航班信息数据结构

**C语言描述如下：**

struct plane //定义结构体数组

{

int num; //定义航班号

char start[30]; //航班起始站

char destination[30]; //终点站

char TakeOffTime[30]; //起飞时间

char ArriveTime[30]; //到达时间

int count; //剩余机票数量

int price; //机票价格

} p[MAXFLIGHT];

2.1.2 客户订单信息

为了便于查找，本订单信息其采用顺序表存储结构，每个数据元素包括订单号、订票用户身份证、订票客户姓名、航班号、订票数量、订单费用六个数据项：

|  |  |  |
| --- | --- | --- |
| 数据项 | 数据类型 | 数据说明 |
| OrderNum | 整型(int) | 订单号 |
| ClientID | 整型(long long) | 订票用户身份证 |
| ClientName | 字符串(char[20]) | 订票客户姓名 |
| FlightNum | 整型(int) | 航班号 |
| TicketNum | 整型(int) | 降落时间 |
| cost | 整型(int) | 订单费用 |

表2 客户订单信息数据结构

**C语言描述如下：**

struct OrderNode

{

int OrderNum; //订单号

long long ClientID; //订票用户身份证

char ClientName[20]; //订票客户姓名

int FlightNum; //航班号

int TicketNum; //购买票数

int cost; //订单费用

} OrderList[E];

2.2 各功能模块调用

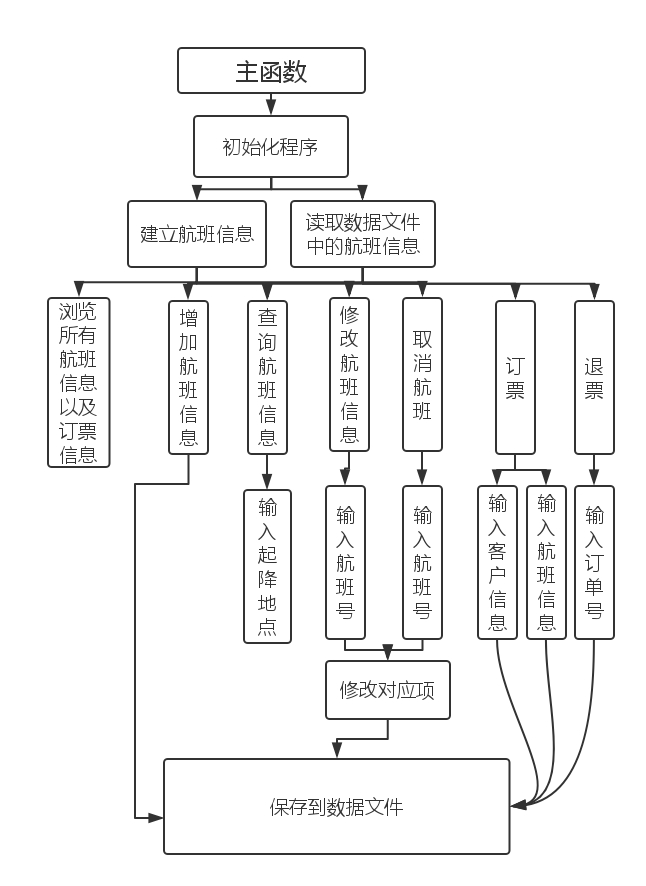


图2.函数调用流程图

2.3 函数说明

本课程设计由如下十三个函数（不计main函数）构成，共有747行代码组成,以及两个数据文件：ClientOrder.txt以及Flight.txt

void add(); //函数声明增加航班信息函数

void print(); //显示航班信息

int\* flightSearch(); //查找航班信息

void book(); //订票业务

void refund(); //退票

void read(); //读取文件

void save(); //保存文件

void output(); //输出格式

void initiate(); //系统初始化

void build(); //建立数据文件

void del(int ); //删除订单

void modify(); //更改航班信息

void cancel(); //删除航班

3. 详细设计

3.1 添加航班信息

在主函数中输入“1”，调用 void add()函数，此函数利用do while循环，用户可持续添加航班信息。CorrectInput变量保证scanf()函数成功接收正确格式的输入数据(航班号、机票数、机票价格等)。在判断输入航班号与数据文件中读取到的航班号无重复后，将其加入到数据文件中Flight.txt保存。并且接受其余数据（起飞地点、降落地点、起飞时间、降落时间等）。流程图如下：

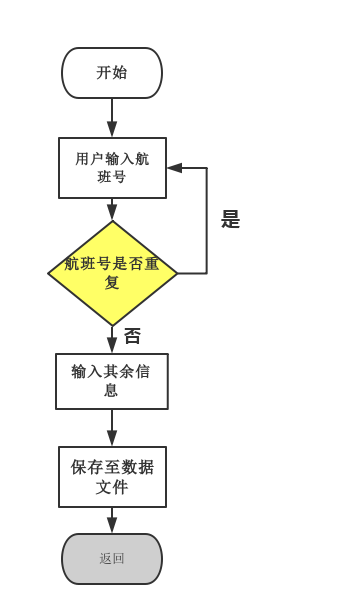


图3. add()函数程序框图

3.2 浏览航班以及订票信息

在主函数中输入“2”，调用 void print()函数完成航班信息以及订票信息的打印。print()首先调用output()函数，该函数利用定义的宏格式输出航班信息以及订单信息。

（即#define PRINT "%-8d %-8s %-8s %-20s %-20s %-5d %-8d\n",

p[i].num,p[i].start,p[i].destination,p[i].TakeOffTime,p[i].ArriveTime,p[i].count,p[i].price

#define FLIGHT "航班号 起始站 终点站 起飞时间 降落时间 机票数 机票价格\n"

#define PRINTORDER "%-9d %-20lld %-8s %-8d %-5d %-8d\n",

OrderList[CurrentOrder].OrderNum,OrderList[CurrentOrder].ClientID,OrderList[CurrentOrder].ClientName,OrderList[CurrentOrder].FlightNum,OrderList[CurrentOrder].TicketNum,OrderList[CurrentOrder].cost

#define ORDER "订单号 客户身份ID 客户名字 航班号 所定票数 订单价钱\n"）

然后利用两个for循环历遍两个顺序表，分别输出对应数据项的值。

3.3 查找航班信息模块

在主函数中输入“3”，调用int\* flightSearch()函数完成航班信息的查询。提示用户输入航班的起飞地点以及降落地点，若在数据文件中找不到此航班，即输出“对不起，没有找到你所需要的信息”，并且提示用户是否需要再次进行查询(Y/N)；若查询成功，则输出所有满足该起飞地点以及降落地点的航班（即输出所有同一航线的航班信息），提示共有多少条满足条件的信息。程序框图如下：

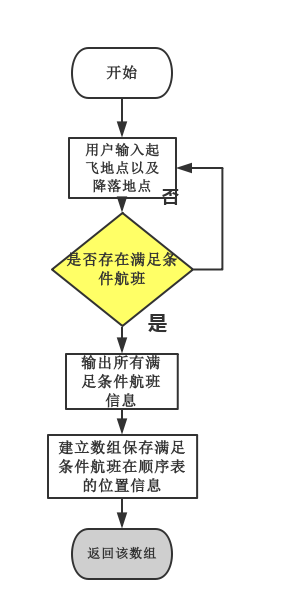


图4. flightSearch()函数程序框图

3.4 客户订票模块

在主函数中输入“4”，调用 void book()函数完成客户的订票。首先调用查询模块，根据输入的起飞地点以及降落地点确定用户需要的航班。假如找不到，则提示不能进行订票；否则，允许用户输入所要定的票数。接着进行分析，若票数小于等于0，则提示不能进行订票，票数至少为1；若票数大于该航班剩余票数，则输出错误，不能进行订票；若该航班剩余票数为0，则推荐该航线其余航班信息供用户选择，可进行其他航班的订票；最后，若满足票数要求，则要求用户输入个人信息（身份证ID以及购票者名字），并且加上由系统生成的订单号，组成一个订单信息，调用save(),保存入另一个数据文件ClientOrder.txt。该模块程序框图如下：

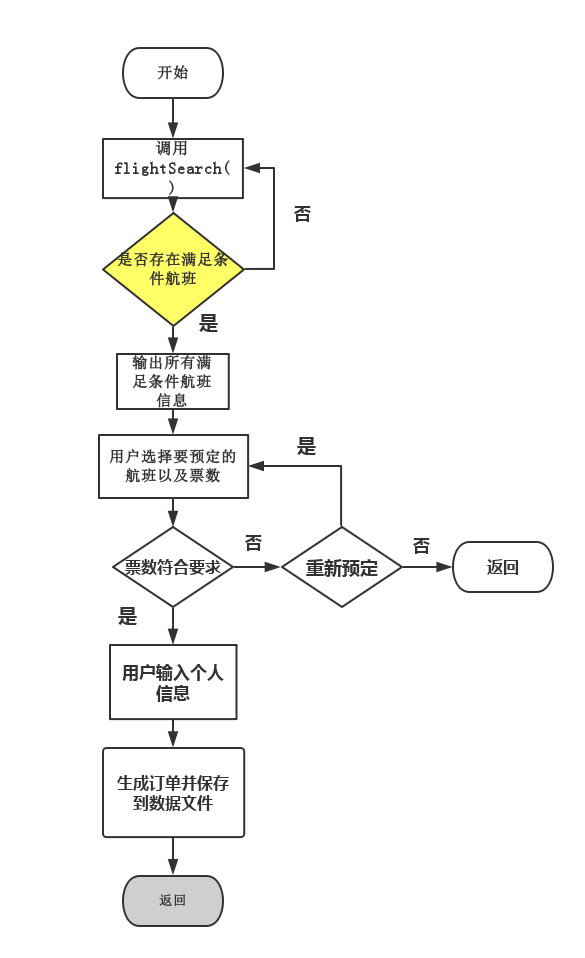


图5. book()函数程序框图

3.5 退票模块

在主函数中输入“5”，调用 void refund()函数完成客户的订票。通过用户输入的订单号码，历遍ClientOrder.txt存储的顺序表，找到该订单，进行顺序表元素的删除操作。并且输出删除订单信息，以及退款总额。最后将修改的结果通过调用save()函数保存到数据文件。

3.6 修改航班信息模块

在主函数中输入“6”，调用 void modify()函数完成航班信息的修改。首先，用户输入要修改的航班号，从而确定要修改的航班。若航班不存在，则提示输入错误，提示是否重新输入；若找到该航班，则用户可以选择1.修改起降点;2.修改时间;3.价格;4.航班号;5.所剩票数0.退出。最后将修改的信息，调用save(),同步到数据文件Flight.txt中。该模块程序框图如下：

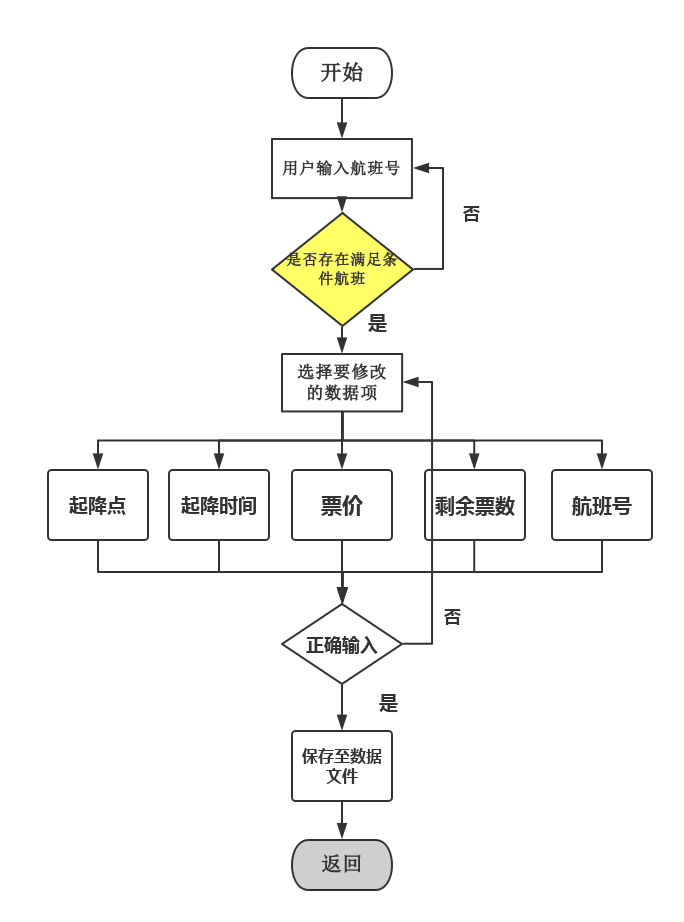


图6. modify()函数程序框图

3.7 航班信息取消模块

在主函数中输入“7”，调用 void cancel()函数完成航班信息的删除。首先，调用flightSerach(),用户输入正确的航班起降地点，确定要删除航班信息在Flight.txt数据文件所保存的顺序表中的位置，再进行删除操作，对该项元素后的元素采取前移操作。最后也是调用save()进行保存。

4. 调试分析



图7. 程序菜单



图8. 添加航班信息



图9. 打印所有航班信息



图10. 打印所有订单信息



图11. 查询航班信息（输出所有满足条件的航班）



图11. 订票功能（若无空位则推荐同一航线其余有空位的航班）



图11. 订票功能（成功订票，输入客户信息，输出支付金额）



图13. 订票功能（订票失败，输入错误）



图13. 退票功能（成功退票，输入退票数，输出退票金额）



图14. 退票功能（退票失败，没有找到该订单）



图15. 退票功能（退票成功，直接删除该订单）

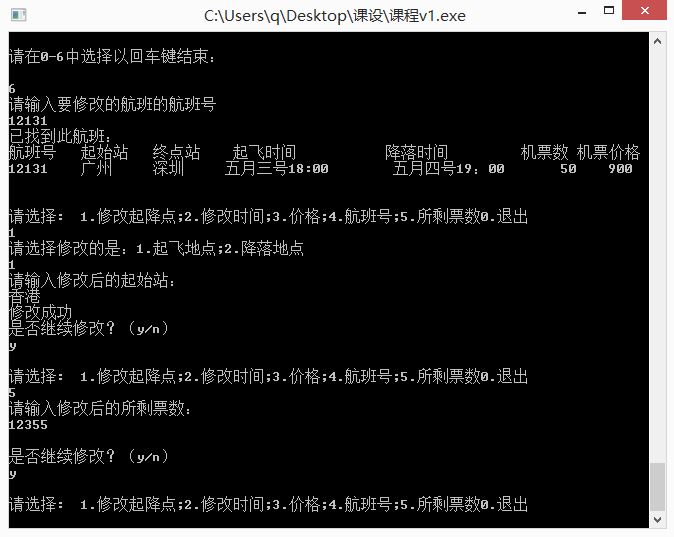


图16. 修改航班信息功能（可修改七个数据项）

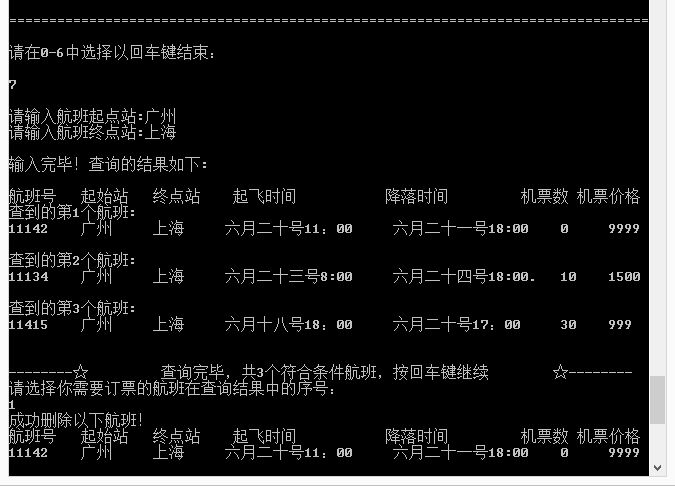


图17. 删除航班信息功能（可选择删除同一航线中一个航班）

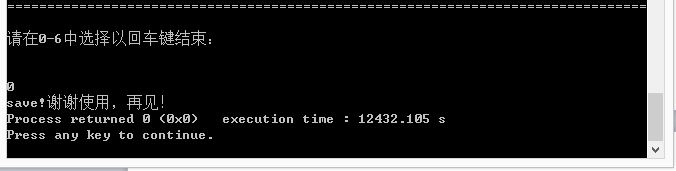


图18. 退出程序并保存数据文件

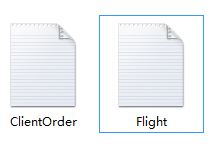


图19. 分别生成两个数据文件

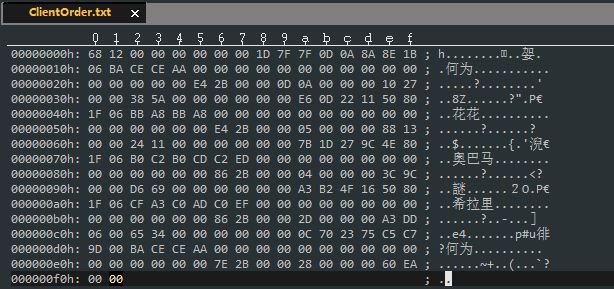


图20. 16进制查看数据文件ClientOrder.txt



图21. 错误输入报错，体现鲁棒性



图22. 错误输入格式，体现鲁棒性

5. 总结

首先，本次实验以及实验报告都是由本人独立完成，也是本人第一次写这么大量的代码，因此感悟很深，也是受益匪浅。先从思路来说吧，其实一开始我打算用单链表或者特殊的邻接表来实现此订票系统，但是由于涉及数据文件以及文件流的标准输入输出问题，所以方便起见选择了顺序表作为存储的数据结构。后续有空会试着用不同的数据结构来尝试实现本系统，也会学一下数据库的链接以及图形化界面的结合使用，可能会选择用JAVA来写。也算是对本系统的一些设想以及提升吧！

然后，自己独立实验的过程，表示有一些小bug也会让你耗费一整天来思考出自哪里，当自己真正解决了会真的有一种成就感，这可能就是抄袭网上的代码与自己写代码的区别吧。但是终究自己能力还是不足，感觉代码量还是比较少，不像计算机专业的同学那样有各种课程设计的机会，可能我们信科只有很少机会做课程设计。我觉得以后也要利用好每一次课程设计的机会进行自己学习，给自己点压力才能做的好。

接着，在本实验中，我也有一些小错误，就是把重心放在数据文件的实现以及同步还有程序的鲁棒性上，可能应该更关心的是数据结构的使用才对，今后会尝试改进并且自己思考下。从设计数据结构到程序各功能模块的实现，再到最后的数据的测试，完全由自己一手一脚完成，与原本的设想不同，本实验其实可以涉及很多优化的数据结构问题，如何使增删改查的操作更加节省时间，就是时间复杂度的分析是需要考虑的。以及一些航班航线优劣的推荐，最小路径问题等也是可以应用到本系统中，但是可能是课堂上的知识缺少实验来支撑，显得有点生疏。

最后，也是挺感谢老师的，带我们入了个门，数据结构这科要学好也是不简单的，以后也会争取时间把由于课时缩减而删掉的部分知识点自己好好补一补。有空也自己动动手完成一些小课程设计类似的程序，多动手才能为以后的积累一定的打码经验，也有利于以后职业生涯的进步。

6. 附录

#include<stdio.h> //标准输入、输出头文件

#include<stdlib.h> //exit()函数

#include<string.h> //包含字符串函数处理头文件

#include<process.h> //包含access函数的头文件

#include<time.h> //随机种子

#define MAXFLIGHT 1000 //定义最多的航班数

#define E 1000 //定义最多的订单数

#define PRINT "%-8d %-8s %-8s %-20s %-20s %-5d %-8d\n",p[i].num,p[i].start,p[i].destination,p[i].TakeOffTime,p[i].ArriveTime,p[i].count,p[i].price //宏定义输出格式

#define FLIGHT "航班号 起始站 终点站 起飞时间 降落时间 机票数 机票价格\n"

#define PRINTORDER "%-9d %-20lld %-8s %-8d %-5d $%-8d\n",OrderList[CurrentOrder].OrderNum,OrderList[CurrentOrder].ClientID,OrderList[CurrentOrder].ClientName,OrderList[CurrentOrder].FlightNum,OrderList[CurrentOrder].TicketNum,OrderList[CurrentOrder].cost

#define ORDER "订单号 客户身份ID 客户名字 航班号 所定票数 订单价钱\n"

struct plane //定义结构体数组

{

int num; //定义航班号

char start[30]; //航班起始站

char destination[30]; //终点站

char TakeOffTime[30]; //起飞时间

char ArriveTime[30]; //到达时间

int count; //剩余机票数量

int price; //机票价格

} p[MAXFLIGHT];

struct OrderNode

{

int OrderNum; //订单号

long long ClientID; //订票用户身份证

char ClientName[20]; //订票客户姓名

int FlightNum; //航班号

int TicketNum; //购买票数

int cost;

} OrderList[E];

int i,m=0,e=0; //定义全局变量i为已读取元素个数

int CurrentOrder,TotalOrder=0;

char again[10];

int result[E];

int IniResult;

int CorrectInput;

void add(); //函数声明增加航班信息函数

void print(); //显示航班信息

int\* flightSearch(); //查找航班信息

void book(); //订票业务

void refund(); //退票

void read(); //读取文件

void save(); //保存文件

void output(); //输出格式

void initiate(); //系统初始化

void build(); //建立数据文件

void del(int ); //删除订单

void modify(); //更改航班信息

void cancel(); //删除航班

void main()

{

int select;

initiate(); //系统初始化判断是否存在原始数据文件

srand((int)time(0));

printf(" ★---您好，欢迎进入飞机订票系统！---★\n");

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

printf("m=%d\n",m);

printf("i=%d\n",i);

printf("TotalOrder=%d\n",TotalOrder);

printf("CurrentOrder=%d\n",CurrentOrder);

do

{

printf("\n --------☆ 1.增加航班信息 ☆-------- \n\n"

" --------☆ 2.浏览航班以及订票信息 ☆-------- \n\n"

" --------☆ 3.查找航班信息 ☆-------- \n\n"

" --------☆ 4.订票业务 ☆-------- \n\n"

" --------☆ 5.退票业务 ☆-------- \n\n"

" --------☆ 6.修改航班信息 ☆-------- \n\n"

" --------☆ 7.航班取消更新 ☆-------- \n\n"

" --------☆ 0.退出 ☆-------- \n\n");

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

printf("请在0-6中选择以回车键结束：\n\n");

scanf("%d",&select);

switch(select)

{

case 1:

add(); //调用增加航班函数

save();

break;

case 2:

print(); //调用显示模块

break;

case 3:

flightSearch(); //调用查找模块

break;

case 4:

book(); //调用订票模块

save();

break;

case 5:

refund(); //调用退票模块

save();

break;

case 6:

modify(); //调用修改航班信息模块

break;

case 7:

cancel(); //调用取消航班信息模块

save();

break;

case 0: //退出系统

save();

printf("谢谢使用，再见！ ");

break;

}

}

while(select!=0); //判断是否调用其他函数

}

void initiate() //定义系统初始化函数

{

if(access("Flight.txt",0)==-1)//如果文件存在，返回0，不存在，返回-1。

build();

else

read();

}

void build() //定义建立数据文件函数

{

// printf("Build函数！！！！");

FILE \*fp,\*fp2; //定义文件指针

if((fp2=fopen("ClientOrder.txt","wb"))==NULL) //打开文件并判定是否出错

{

printf("创建订票信息文件失败! "); //打印出错提示

getchar();

return;

}

fclose(fp2);

if((fp=fopen("Flight.txt","wb"))==NULL) //打开文件并判定是否出错

{

printf("创建航班信息文件失败! "); //打印出错提示

getchar();

return;

}

printf("请依次输入航班信息(以回车键结束):\n"); //打印提示信息

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

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

{

printf("请输入航班号: ");

CorrectInput=scanf("%d",&p[i].num); //输入航班号

while(CorrectInput==0)

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&p[i].num);

}

printf("请输入起始站: ");

scanf("%s",p[i].start); //输入起始站

printf("请输入终点站: ");

scanf("%s",p[i].destination); //输入终点站

printf("请输入起飞时间: ");

scanf("%s",p[i].TakeOffTime); //输入起飞时间

printf("请输入降落时间: ");

scanf("%s",p[i].ArriveTime); //输入降落时间

printf("请输入机票数: ");

CorrectInput=scanf("%d",&p[i].count); //输入机票数

while(CorrectInput==0)

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&p[i].count);

}

printf("请输入票价：");

CorrectInput=scanf("%d",&p[i].price);

while(CorrectInput==0)

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&p[i].price);

}

fwrite(&p[i],sizeof(struct plane),1,fp);//模块写入数据文件

m++; //总航班数加1

printf("添加完毕,是否继续添加?请键入y或n以回车键结束:");

scanf("%s",again);

if(strcmp(again,"y")!=0) //判断是否继续添加航班信息

{

fclose(fp); //关闭文件

return;

}

}

}

void read() //定义读取文件函数

{

FILE \*fp,\*fp2;

if((fp=fopen("Flight.txt","r"))==NULL)

{

printf("\n Warning!!读取航班信息文件失败!!!!! \n");

getchar();

exit(1);

}

if((fp2=fopen("ClientOrder.txt","r"))==NULL)

{

printf("\n Warning!!读取订票文件失败!!!!! \n");

getchar();

exit(1);

}

i=0;

while(!feof(fp))

{

fread(&p[i],sizeof(struct plane),1,fp); //逐块读取数据

i++;

m++; //计算存在航班数

}

m--;

fclose(fp);

CurrentOrder=0;

while(!feof(fp2))

{

fread(&OrderList[CurrentOrder],sizeof(struct OrderNode),1,fp2); //逐块读取数据

CurrentOrder++;

TotalOrder++; //计算存在航班数

}

TotalOrder--;

fclose(fp2);

}

void save() //定义保存函数

{

printf("save!");

FILE \*fp,\*fp2;

if((fp=fopen("Flight.txt","w"))==NULL)

{

printf("写入航班信息文件失败! ");

getchar();

return;

return;

}

for(i=0; i<m; i++) //逐块保存数据

fwrite(&p[i],sizeof(struct plane),1,fp);

fclose(fp);

if((fp2=fopen("ClientOrder.txt","w"))==NULL)

{

printf("写入订票信息文件失败! ");

getchar();

return;

return;

}

for(CurrentOrder=0; CurrentOrder<TotalOrder; CurrentOrder++) //逐块保存数据

fwrite(&OrderList[CurrentOrder],sizeof(struct OrderNode),1,fp2);

fclose(fp2);

}

void add() //定义增加航班信息函数

{

int Tempnum,Tempcount,Tempprice;

do

{

printf("请依次输入您要增加的航班信息(以回车键结束): \n"); //打印提示信息

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

printf("请输入航班号: ");

CorrectInput=scanf("%d",&Tempnum);

while(CorrectInput==0) //读取航班号

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&Tempnum);

}

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

{

if(Tempnum==p[i].num)

{

printf("输入错误！存在重复航班号！现返回菜单\n");

return;

}

}

p[m].num=Tempnum;

printf("请输入起始站: ");

scanf("%s",p[m].start); //读取起始站

printf("请输入终点站: ");

scanf("%s",p[m].destination); //读取终点站

printf("请输入起飞时间: ");

scanf("%s",p[m].TakeOffTime); //读取起飞时间

printf("请输入降落时间: ");

scanf("%s",p[m].ArriveTime); //读取降落时间

printf("请输入机票数: ");

CorrectInput=scanf("%d",&p[m].count); //读取机票数

while(CorrectInput==0) //读取航班号

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&p[m].count);

}

printf("请输入价格：");

CorrectInput=scanf("%d",&p[m].price); //读取机票价格

while(CorrectInput==0) //读取航班号

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&p[m].price);

}

m++; //读取总航班信息

printf("添加完毕,是否继续添加?请键入y或n以回车键结束:");

scanf("%s",again);

}

while(!strcmp(again,"y")); //判断是否继续添加

}

void cancel() //取消航班

{

int\* outcome;

int select;

outcome=flightSearch(); //调用查询模块

if(outcome[0]==-1)

{

printf("对不起!没有找到您所需要的航班,所以不能订票。\n"); //未查找到所需航班

printf("\n请按回车键返回上层菜单 ");

getchar();

getchar();

return;

}

if(outcome[1]!=-1)//搜索结果大于1，需要选择一个

{

printf("请选择你需要订票的航班在查询结果中的序号：\n");

scanf("%d",&select);

i=outcome[select-1];

}

else

i=outcome[0];

printf("成功删除以下航班！\n");

printf(FLIGHT);

printf(PRINT);

for(; i<m; i++)

{

p[i]=p[i+1];

}

m--;

}

void output() //定义输出格式函数

{

printf(FLIGHT); //信息标题

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

printf(PRINT);//打印出信息

printf("展示当前所有订单：\n");

printf(ORDER);

for(CurrentOrder=0; CurrentOrder<TotalOrder; CurrentOrder++)

printf(PRINTORDER);

printf("m=%d\n",m);

printf("i=%d\n",i);

printf("TotalOrder=%d\n",TotalOrder);

printf("CurrentOrder=%d\n",CurrentOrder);

}

void print() //定义显示航班信息函数

{

printf("\n目前我们有如下航班：\n");

output(); //调用输出格式函数

printf("\n请按回车键返回上层菜单 ");

getchar();

getchar();

}

int\* flightSearch() //定义查询函数

{

int flag=0;

int count=0;

int j=0;

char a[30];

char b[30];

for(IniResult=0;IniResult<E;IniResult++)

result[IniResult]=-1;

do

{

printf("\n请输入航班起点站:");

scanf("%s",a); //输入查询的航班起点

printf("请输入航班终点站:");

scanf("%s",b);

printf("\n输入完毕！查询的结果如下:\n\n");

printf(FLIGHT);

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

{

if(strcmp(p[i].start,a)==0&&strcmp(p[i].destination,b)==0) //按航班起点和终点站判定输出条件

{

count++;

printf("查到的第%d个航班:\n",count);

printf(PRINT); //显示信息

flag=1;

result[j++]=i;

}

}

if(flag!=0)

{

printf("\n--------☆ 查询完毕，共%d个符合条件航班，按回车键继续 ☆--------",count);

// int k=0;

// printf("result数组：\n");

// for(;k<j;k++)

// printf("%d %d %s %s 星期%s %d %d\n",result[k],p[result[k]].num,p[result[k]].start,

// p[result[k]].destination,p[result[k]].time,p[result[k]].count ,p[result[k]].price);

getchar();

getchar();

return result;

}

else

{

printf("\n--------☆ 对不起，没有您需要的信息！ ☆--------\n");

printf("--------☆ 是否重新查找?请键入y或n以回车键结束");

scanf("%s",again);

}

}

while(!strcmp(again,"y")); //判定是否重新查找

return result;//返回result数组（保存找到的航班对应数组下标）

}

void book() //定义订票业务函数

{

int\* outcome;

int WantToBookNum;

int TempID;

int select;

int CountClient;

int flag=0;

printf("CurrentOrder:%d",CurrentOrder);

do

{

outcome=flightSearch(); //调用查询模块

if(outcome[0]==-1)

{

printf("对不起!没有找到您所需要的航班,所以不能订票。\n"); //未查找到所需航班

printf("\n请按回车键返回上层菜单 ");

getchar();

getchar();

break;

}

do

{

if(outcome[1]!=-1)//搜索结果大于1，需要选择一个

{

printf("请选择你需要订票的航班在查询结果中的序号：\n");

scanf("%d",&select);

i=outcome[select-1];

}

else

i=outcome[0];

printf("请输入您要订的机票数（以回车键结束）： ");

CorrectInput=scanf("%d",&WantToBookNum); //输入所订机票数

while(CorrectInput==0)

{

printf("输入错误!请输入正确格式数据！\n");

getchar();

CorrectInput=scanf("%d",&WantToBookNum);

}

if(WantToBookNum<=0) //判定机票数是否出错

{

printf("输入错误！至少需订1张机票。\n");

}

else if(p[i].count==0)//判定机票是否售完

{

printf("对不起，你所选择的航班的机票已售完!\n");

if(outcome[1]!=-1)//搜索结果大于1，需要选择一个

{

printf("现为你推荐以下航班：\n");

for(IniResult=0;IniResult<E;IniResult++)

if(outcome[IniResult]!=-1&&p[outcome[IniResult]].count!=0)

{

i=outcome[IniResult];

printf(PRINT);

}

}

break;

}

else if(p[i].count!=0&&p[i].count>=WantToBookNum) //判定机票数是否大于等于订票数

{

printf("请输入您的姓名：");

scanf("%s",OrderList[TotalOrder].ClientName);

printf("请输入您的身份证：");

scanf("%lld",&(OrderList[TotalOrder].ClientID));

OrderList[TotalOrder].FlightNum=p[i].num;

OrderList[TotalOrder].TicketNum=WantToBookNum;

OrderList[TotalOrder].OrderNum=rand();

OrderList[TotalOrder].cost=p[i].price\*WantToBookNum;

TotalOrder++;

p[i].count=p[i].count-WantToBookNum;

printf("订票成功!\n");

printf("请支付$%d元，谢谢！\n",p[i].price\*WantToBookNum);

break;

}

else if(p[i].count<WantToBookNum)//判定机票数是否小于订票数

{

printf("对不起，你所选择的航班只剩%d张机票\n", p[i].count);

printf("要重新输入要预定的机票数吗?请输入y或n以回车键结束: ");//判定是否重新输入订票数

scanf("%s",again);

}

}

while(!strcmp(again,"y"));

printf("\n是否需要订其他航班的机票?请输入y或n以回车键结束: ");

scanf("%s",again);

}

while(!strcmp(again,"y")); //判定是否继续订票

}

void refund() //定义退票函数

{

int\* h;

int n;

int flag=0;

int num;

int temp;

do

{

// printf("请输入退票的订单号或者");

printf("请输入退票的订单号：\n");

scanf("%d",&num);

for(CurrentOrder=0; CurrentOrder<TotalOrder; CurrentOrder++)

{

if(OrderList[CurrentOrder].OrderNum==num)

{

flag=1;

printf("已找到该订单,请选择：\n");

printf("y:订单取消;n:退掉部分票（y/n）\n");

scanf("%s",again);

if(strcmp(again,"y"))

{

printf("请问要退多少张票呢:\n");

scanf("%d",&n);

if(n<=0)

{

printf("输入错误，至少退掉一张票!\n");

return;

}

else if(n>OrderList[CurrentOrder].TicketNum)

printf("输入错误，退票多于已定的票！\n");

else if(n==OrderList[CurrentOrder].TicketNum)

{

printf("删除订单！\n");

del(CurrentOrder);

}

else

{

temp=(OrderList[CurrentOrder].cost/OrderList[CurrentOrder].TicketNum)\*n;

printf("成功退了%d张票,共退回%d元！谢谢\n",n,temp);

OrderList[CurrentOrder].TicketNum-=n;

OrderList[CurrentOrder].cost-=temp;

}

}

else

{

printf("删除订单！\n");

del(CurrentOrder);

}

}

}

if(flag==0)

{

printf("对不起!没有找到您所需要的航班,所以不能退票。\n");

printf("\n请按回车键返回上层菜单 ");

getchar();

getchar();

break;

}

printf("是否继续退其他票？ 请键入y或n以回车键结束: "); //判定是否继续退票

scanf("%s",again);

}

while(!strcmp(again,"y")); //判定并跳出循环

}

void del(int delOrder)//删除订单

{

int a,j,n;

for(a=delOrder; a<TotalOrder; a++)

{

OrderList[a]=OrderList[a+1];

}

printf("删除成功！\n");

TotalOrder--;

}

void modify()//改变航班信息

{

int ModifyOption,CheckPnum,Startordestination,w,q;

int TakeOffOrArrive;

int remain;

char h[30],g[30],again[20];

int flag=0;

printf("请输入要修改的航班的航班号\n");

scanf("%d",&CheckPnum);

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

{

if(p[i].num==CheckPnum)

{

printf("已找到此航班：\n");

flag=1;

printf(FLIGHT);

printf(PRINT);

do

{

printf("\n请选择: 1.修改起降点;2.修改时间;3.价格;4.航班号;5.所剩票数0.退出\n");

scanf("%d",&ModifyOption);

switch(ModifyOption)

{

case 1:

printf("请选择修改的是：1.起飞地点;2.降落地点\n");

scanf("%d",&Startordestination);

if(Startordestination==1)

{

printf("请输入修改后的起始站：\n");

scanf("%s",h);

strcpy(p[i].start,h);

printf("修改成功");

}

else if(Startordestination==2)

{

printf("请输入修改后的终点站：\n");

scanf("%s",h);

strcpy(p[i].destination,h);

printf("修改成功");

}

break;

case 2:

printf("请选择修改的是：1.起飞时间;2.到达时间\n");

scanf("%d",&TakeOffOrArrive);

if(TakeOffOrArrive==1)

{

printf("请输入修改后的时间\n");

scanf("%s",g);

strcpy(p[i].TakeOffTime,g);

printf("修改成功");

}

else if(TakeOffOrArrive==2)

{

printf("请输入修改后的时间\n");

scanf("%s",g);

strcpy(p[i].ArriveTime,g);

printf("修改成功");

}

break;

case 3:

printf("请输入修改后的价格\n");

scanf("%d",&w);

p[i].price=w;

printf("修改成功");

break;

case 4:

printf("请输入修改后的航班号\n");

scanf("%d",&q);

p[i].num=q;

break;

case 5:

printf("请输入修改后的所剩票数：\n");

scanf("%d",&remain);

p[i].count=remain;

break;

case 0: //退出系统

return;

}

printf("\n是否继续修改？（y/n）\n");

scanf("%s",again);

}

while(!strcmp(again,"y"));

save();

}

}

if(flag==0)

printf("航班号输入错误！！并没有这航班！现退出至菜单\n");

}

参考文献

[1]严蔚敏,李冬梅,吴伟民. 数据结构(C语言版)[J]. 计算机教育,2012,12:62.