第三篇

课程设计实验

实验 1　航空客运订票系统

【系统实现】 航空客运订票的业务活动包括查询航线和客票预订的信息、客票预订和办理退票等。设计一个计算机程序以使上述任务借助计算机来完成。

【基本要求】

1. 系统必须存储的数据信息

(1) 信息：飞机抵达城市、航班号、飞机号、起降时间、航班票价、票价折扣、总位置和剩余位置、已订票的客户名单。

(2) 客户信息：客户姓名、证件号、座位号。

2. 系统能实现的操作和功能

(1) 承办订票业务：根据客户提出的要求(飞机抵达城市、起降时间、订票数量)查询该航班信息(包括票价、折扣和剩余位置)，若满足要求，则为客户办理订票手续，输出座位号。

(2) 承办退票业务：根据客户提供的情况(航班号、订票数量)，为客户办理退票手续。

(3) 查询功能。

① 查询航线信息。根据飞机降落地点，输出下列信息：航班号、飞机号、起降时间、航班票价、票价折扣和剩余位置。

② 查询客户预订信息。根据客户证件号，输出下列信息：航班号、飞机号和座位号。

【实现提示】

1. 设计思想

航班信息和客户预订名单采用线性表表示，为了插入和删除方便，建议以链表作为存储结构。整个系统需要汇总各条航线的情况并登录在一张线性表上，考虑到航线基本不变，可采用顺序存储结构，并按航班有序或按终点站名有序。每条航线是这张表上的一个记录，包括上述 9 个域，其中乘员名单域为指向乘员名单链表的头指针。

2. 参考程序

系统采用菜单的方式由操作者进行选择，并初始化了航线和预订顾客的信息，作为示意系统，全部数据可以只放在内存中。

```
#include<stdio.h>
#include<malloc.h>
#include<string.h>
#define OK 1
#define ERROR 0
typedef struct airline{                                    /* 飞机航班的结构定义 */
        char air_num[8];
        char plane_num[8];
        char end_place[20];
        int total;
        int left;
        struct airline *next;
}airline;
typedef struct customer{                                   /*顾客信息的结构定义 */
        char name[8];
        char air_num[8];
        int seat_num;
        struct customer *next;
}customer;
airline *start_air()                                       /*创建航班链表 */
{
        airline *a;
        a=(airline*)malloc(sizeof(airline));
        if(a==NULL)
       printf("空间不足");
        return a;
}
customer *start_cus()                                      /*创建顾客链表 */
{
        customer *c;
        c=(customer*)malloc(sizeof(customer));
        if(c==NULL)
       printf("空间不足");
        return c;
}
airline *modefy_airline(airline *l,char *air_num)          /* 修改航班的空余座位信息 */
{
        airline *p;
        p=l->next;
        for(;p!=NULL;p=p->next)
        {
                if(strcmp(air_num,p->air_num)==0)
                {
                        p->left++;
                        return l;
                }
                printf("NO the airline!");
                return 0;
        }
```

```
}
int insert_air(airline **p,char *air_num,char *plane_num,char *end_place,int total,
int left)                                                    /* 增加航班信息 */
{
        airline *q;
        q=(airline*)malloc(sizeof(airline));
        strcpy(q->air_num,air_num);
        strcpy(q->plane_num,plane_num);
        strcpy(q->end_place,end_place);
        q->total=total;
        q->left=left;
        q->next=NULL;
        (*p)->next=q;
        (*p)=(*p)->next;
        return OK;
}
int insert_cus(customer **p,char *name,char *air_num,int seat_num)
                                                             /*增加某航班的顾客信息*/
{
        customer *q;
        q=(customer*)malloc(sizeof(customer));
        strcpy(q->name,name);
        strcpy(q->air_num,air_num);
        q->seat_num=seat_num;
        q->next=NULL;
        (*p)->next=q;
        (*p)=(*p)->next;
        return OK;
}
int book(airline *a,char *air_num,customer *c,char *name)              /*订票操作*/
{
        airline *p=a;
        customer *q=c->next;
        p=a->next;
        for(;q->next!=NULL;q=q->next){}
        for(;p->next!=NULL;p=p->next)
        {
                if(p->left>0)
                {
                        printf("Your seat number is %d",(p->total-p->left+1));
                        insert_cus(&q,name,air_num,p->total-p->left+1);
                        p->left--;
                        return OK;
                }
                else
                {
                        printf("seat is full");
                        return 0;
                }
```

```
        }
}

int del_cus(customer *c,airline *l,char *name)          /*取消订票信息操作*/
{
        customer *p, *pr;
        char air_num[8];
        pr = c;
        p = pr->next;
        while(p! = NULL)
        {
                if(strcmp(p->name,name) == 0)
                {
                        strcpy(air_num,p->air_num);
                        l = modefy_airline(l,air_num);
                        pr->next = p->next;
                        p = pr->next;
                        printf("finish!");
                        return OK;
                }
                pr = pr->next;
                p = pr->next;
        }
        printf("NO the customer!");
        return ERROR;
}
int search_air(airline *head)                           /*查找航班信息操作*/
{
        airline *p = head->next;
        printf("air_num plane_num end_place total left\n");
        for(;p! = NULL;p = p->next)
        {
          printf("%s%-10s%-8s   %-8d%-8d\n",p->air_num,p->plane_num,p->
          end_place,p->total,p->left);
        }
        return OK;
}
int search_cus(customer *head)                          /*查找顾客信息操作*/
{
        struct customer *q = head->next;
        printf("name  air_num  seat_num\n");
        for(;q! = NULL;q = q->next)
        {
                printf("%-8s%-12s%-d\n",q->name,q->air_num,q->seat_num);
        }
        return OK;
}
int creat_air(airline **l)                              /*预先设置航班信息*/
{
```

```
        airline *p = *l;
        int i = 0;
        char *air_num[3] = {"007af","008af","009af"};
        char *plane_num[3] = {"plane1","plane2","plane3"};
        char *end_place[3] = {"Beijing","Shanghai","Tianjin"};
        int total[3] = {100,100,100};
        int left[3] = {52,54,76};
        for(i = 0;i < 3;i++)
        insert_air(&p,air_num[i],plane_num[i],end_place[i],total[i],left[i]);
        return OK;
}
int creat_cus(customer **l)                              /* 预先设置已订票的顾客信息 */
{
        customer *p = *l;
        int i = 0;
        char *name[3] = {"zhangsan","lisi","wangwu"};
        char *air_num[3] = {"007af","008af","009af"};
        int seat_num[3] = {2,5,7};
        for(i = 0;i < 3;i++)
        insert_cus(&p,name[i],air_num[i],seat_num[i]);
        return OK;

}
void main()
{
        int t = 1;
        customer *cus = start_cus();
        airline *air = start_air();
        char name[8],air_num[8],ch;
        creat_air(&air);
        creat_cus(&cus);
        while(t == 1)
        {
                printf("\n");
                printf("*********************************\n");
                printf("*   Welcome to air firm!    *\n");
                printf("*     book --------- 1      *\n");
                printf("*     cancel ------ 2       *\n");
                printf("*     search ------ 3       *\n");
                printf("*     exit --------- 4      *\n");
                printf("*********************************\n");
                ch = getch();
                if(ch == '1')
                {
                        printf("Please input a airline number:");
                        scanf("%s",air_num);
                        printf("Please input a name:");
                        scanf("%s",name);
                        book(air,air_num,cus,name);
```

```
                    }
                    else
                        if(ch == '2')
                        {
                            printf("Please input the cancel name:");
                            scanf(" % s",name);
                            del_cus(cus,air,name);
                        }
                        else
                            if(ch == '3')
                            {
                                    search_air(air);
                                    printf("\n");
                                    search_cus(cus);
                            }
                            else
                                if(ch == '4')
                                {
                                    t = 0;
                                }
        }
}
```

3. 程序完善

(1) 上述参考程序实现了哪些功能？指出存在的缺陷。

(2) 程序如何完善？

① 功能上的完善。

② 编写结构上的完善。

③ 程序控制上的完善。

实验 2　汉诺塔游戏程序

【系统实现】 Hanoi(汉诺)塔问题是指：在平面上有三个位置 a、b、c，在 a 位置上有 n 个大小不等的圆盘，小盘压在大盘之上。要求将 a 位置的 n 个圆盘，通过 b 位置移动到 c 位置上，并仍按同样的顺序叠放。移动圆盘时必须遵循下列规则。

(1) 每一次只能够移动一个圆盘；

(2) 圆盘可以插放在 a、b 和 c 中任何一个塔座上；

(3) 任何时刻都不能将一个较大的圆盘压在较小的圆盘之上。

要解决以上问题的基本思想如下：

当 n＝1 时，问题简单，只要将该圆盘从塔座 a 上移动到塔座 c 上即可。

当 n＞1 时，需要利用塔座 b 作为辅助塔座，若能设法将除底下最大的圆盘之外的 n－1个圆盘从塔座 a 移动到塔座 b 上，就可以将最大的圆盘从塔座 a 移至塔座 c 上，然后再将塔座 b 上的其余 n－1 个圆盘移动到塔座 c 上。而如何将 n－1 个圆盘从一个塔座移至另一个塔座，则跟原问题具有相同的特征属性，只是问题的规模小于 1，因此可以用同样的方法求解。

【基本要求】

(1) 要求用盘的个数可选择，即可从键盘输入(如 3～64 等)。

(2) 用动画的方式在屏幕上显示盘的移动路线。

(3) 完成的程序界面要美观，能够完成游戏的整个过程。

【实现提示】

(1) 根据问题描述，可以将问题解决分为三步。

① 将 a 位置的 N－1 个通过 c 位置移动到 b；

② 将 a 位置的最后一个直接移到 c 位置；

③ 将 b 位置的 N－1 个通过 a 位置移动到 c。完成整个任务。

其中的①和③原理完全一样，只是位置不同，而①或③的完成，又可看做初始问题的解决。以③为例，即在上面移动的基础上，将 b 位置的 N－2 个通过 c 移动到 a，再将 b 位置的 N－1 个直接移动到 c，再将 a 位置的 N－2 个移动到 c 位置。

其余的以此类推，直到 N＝1 时为止。

以上的过程，正是 C 语言函数中的递归调用。

(2) 问题的数学描述。

依据上面的分析，可以构造两个函数来解决这个问题，一个函数为：

```
hanoi(int n,char a,char b,char c),
```

产生递归调用,其中n为移动的个数,a、b、c分别为从一个位置、经过另一个位置、到其他位置这三个位置。

另一个就是移动函数:

```
move(char a,char c),
```

表示从一个位置移动到另一个位置,a和c同上。其递归调用关系如下所述:

```
void Hanoi ( int n, char a, char b, char c )
/*将塔座a从上到下编号为1～n,且将按直径由小到大叠放的n个圆盘按规则移动到塔座c上,
  塔座b用作辅助塔座 */
   {  if( n= =1 )
        move ( a, c ) ;              /*将编号为1的圆盘从塔座a移动到塔座c*/
      else
       {  Hanoi ( n-1, a, c, b );  /*将a上编号为1～n-1的圆盘移至b、c并作辅助塔座*/
          move ( a, c );           /*将编号为n的圆盘从a移动到c*/
          Hanoi ( n-1, b, a, c );  /*将b上编号为1～n-1的圆盘移至c、a并作辅助塔座*/
       }
   }
```

(3) 所需头文件。

```
#include "graphics.h"
#include "dos.h"
#include "stdio.h"
#include "alloc.h"
#include "stdlib.h"
#include "conio.h"
```

(4) 显示与隐藏图像常用函数。

```
size=imagesize(x1,y1,x2,y2);                 /*计算屏幕图像的大小 */
buffer=malloc(size);                         /*申请用于保存图像的内存 */
getimage(x1,y1,x2,y2,buffer);                /*保存图像进入缓冲区 */
putimage(xm1,ym1,buffer,COPY_PUT);           /*从缓冲区中恢复图像 */
free(buffer);                                /*释放内存缓冲区 */
```

【程序实现】

```
#include "graphics.h"
#include "dos.h"
#include "stdio.h"
#include "alloc.h"
int num1,num2,num3,h0,cy[66];
void plot1(int ,int ,int ,int );
void move(char getone,char putone)
{  int x0=40,x,y,w,h,tx,ty,tw,th,x1,x2,y1,y2,xm1,ym1,xm2,ym2;
   int i,n,size;
   void *buffer,*buffer1;
   switch(getone)
   {
```