

目录

1	程序设计概要	4
1.1	设计目标.....	4
1.2	总体设计.....	4
1.3	程序功能逻辑设计	4
1.4	程序模块逻辑框图.....	4
2	程序详细解析	5
2.1	头文件.....	5
2.2	宏定义.....	5
2.3	函数声明.....	5
2.4	声明结构体类型 bookk.....	5
2.5	声明结构体类型 idd.....	6
2.6	声明结构体类型 ree.....	6
2.7	定义变量.....	6
2.8	定义 count()函数——读取文件中的书籍信息写入结构体数组.....	7
2.9	定义 id_read()函数——读取文件中的账户信息写入结构体数组	7
2.10	定义 re_read()函数——读取文件中的借书信息写入结构体数组.....	8
2.11	定义 file_w ()函数——将书籍结构体的信息写入文件	8
2.12	定义 clear()函数——整理空书籍.....	9
2.13	定义 list_print()函数——输出书的列表.....	9
2.14	定义 refile_w()函数——将还书结构体的信息写入文件	10
2.15	定义 sign()函数——登录页.....	10
2.16	定义 menu()函数——菜单页	13
2.17	定义 menu_yk()函数——游客菜单页	15
2.18	定义 list()函数——书籍清单页	17
2.19	定义 list_yk()函数——游客书籍清单页	17
2.20	定义 borrow()函数——借书.....	18

2.21 定义 ret()函数——还书	19
2.22 定义 update()函数——修改图书信息	20
2.23 主函数	20
3 使用说明	21
3.1 启动系统	21
3.2 输入	21
3.3 输出	21
4 运行测试	22
5 问题及解决办法、心得体会	24

1 程序设计概要

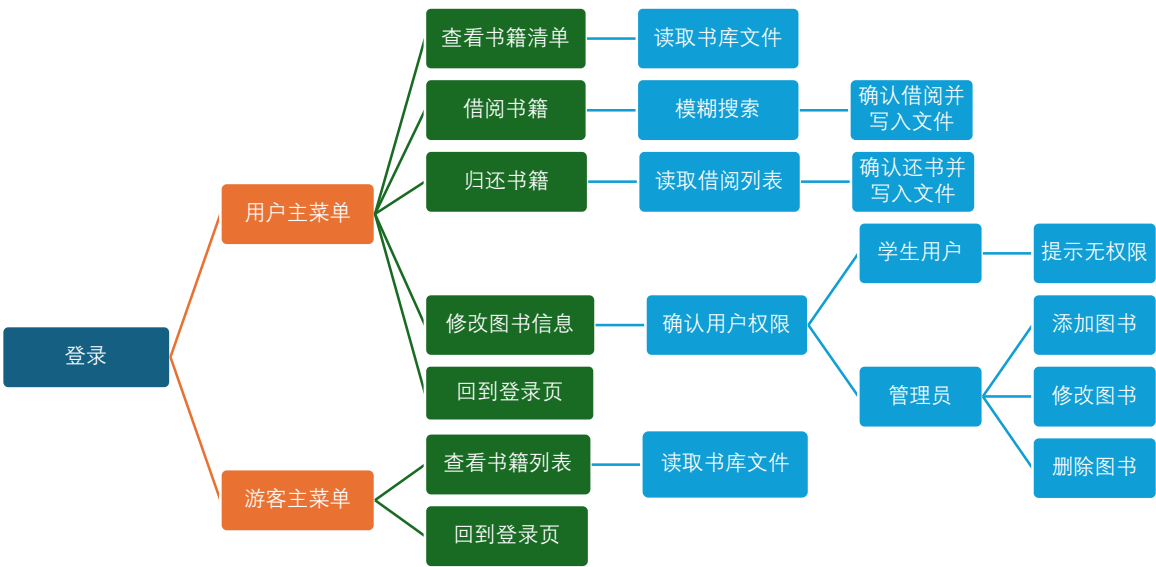
1.1 设计目标

创建一个简单的图书馆管理系统，用于管理图书、读者和借阅信息，具备①新进图书及其基本信息输入；②图书基本信息查询；③对某些图书信息的删除、修改；④办理借还书手续；⑤用户登录和权限管理（管理员和普通用户）的功能。

1.2 总体设计

该图书管理系统面向学生、管理员以及游客，对于学生和管理员，该系统具备查看书籍清单、借阅书籍、归还图书、修改图书信息和退出账号的功能，其中修改图书信息仅能由管理员操作；对于游客，该系统具备查看书籍清单、回到登录界面以及退出系统的功能。

1.3 程序功能逻辑设计



1.4 程序模块逻辑框图

main																
sign															re_read	count
id_read	menu													menu_yk		
	list	borrow				ret			update				list_yk			
	count	count	file_w	re_read	refile_w	re_read	file_w	refile_w	count	file_w	list	clear	list_print	count		

2 程序详细解析

2.1 头文件

引入了三个头文件——标准输入输出库<stdio.h>、包含 system()函数的<stdlib.h>和处理字符串的<string.h>。

```
#include "stdio.h"
#include "stdlib.h"           //包含 system()函数
#include "string.h"
```

2.2 宏定义

```
#define N 100                //书籍结构体定义数量
#define Nid 50               //账号结构体定义数量
#define Nre 100              //还书结构体定义数量
```

2.3 函数声明

```
//主功能区
void sign();                //登录
void menu();                //菜单主页
void list();                //显示书列表
void borrow();              //借书
void ret();                 //还书
void update();              //修改书信息

void menu_yk();              //游客模式菜单
void list_yk();              //游客模式显示

void count();               //功能函数区
void file_w();               //将书籍结构体的信息写入文件
void clear();               //整理空书籍
void list_print();           //不清除显示内容的情况下输出书的列表
void refile_w();             //将还书结构体的信息写入文件
```

2.4 声明结构体类型 bookk

以书名、作者、出版社、ISBN 号、馆藏数、可借数、日期的顺序存放的书籍信息。随后定义了一个结构体数组，并为数组中的前两个位置提供了书籍的初始数据。

```
struct bookk
{
    char name[100];          //书名
    char author[100];        //作者
    char publisher[100];      //出版社
    char ISBN[20];           //ISBN 号
    int num_z;                //馆藏数
    int num_x;                //可借数
    char date[20];            //日期
```

```
char f_query; //用于全局模糊查询 标记单项是否已出现
};
```

2.5 声明结构体类型 idd

以账号、密码和账户类型的顺序存放的账户信息。随后定义了一个结构体数组，并为数组中的前两个位置提供了账户的初始数据。

```
struct idd
{
    char account[20];
    char password[20];
    int perm;
};
```

2.6 声明结构体类型 ree

以需还书的书名、借书人账号和需还书的类型（用于确保在显示还书记录时，只显示与当前登录用户相关的那些记录，避免显示其他用户的还书信息）的顺序存放的账户信息。随后定义了一个结构体数组，并为数组中的前两个位置提供了需还书的初始数据。

```
struct ree
{
    char retname[100];
    char belong[20];
    char r_query;
};
```

2.7 定义变量

- 1) **bookkinds**: 记录当前系统中书籍的类型数。在 [count\(\)函数](#)中，系统通过读取 booklist.txt 文件来计算书籍数量，并将有效的书籍数量（馆藏数不为 0 的书籍）赋值给 bookkinds。
- 2) **id_num**: 记录当前系统中用户账号的总数。在 [id_read\(\)函数](#)中，系统通过读取 idlist.txt 文件来计算用户账号数量，并将有效的账号数量（权限为 0 或 1 的账号）赋值给 id_num。
- 3) **re_num**: 记录当前系统中需要归还的书籍总数。在 [re_read\(\)函数](#)中，系统通过读取 returnlist.txt 文件来计算需要归还的书籍数量，并将有效的还书记录数量赋值给 re_num。
- 4) **char name_r[100]={0}**: 在借书功能中，name_r 用于存储用户输入的模糊查询关键词，以便在书籍列表中进行搜索。它被初始化为一个长度为 100 的字符数组，并全部填充为 0（即空字符）。
- 5) **int permission=0**: 记录当前登录用户的权限。如果 permission 为 0，表示当前用户为学生用户，只能使用查看书籍清单、借还书。如果

permission 为 1，表示当前用户为管理员用户，可以执行所有功能，包括修改书籍信息。

- 6) **char account_now[20]={0}**: 存储当前登录用户的账号信息。在用户登录成功后，系统将登录账号赋值给 account_now，以便在还书过程中验证用户身份，显示对应的未还书的信息和记录借书归还信息等。

```
static int bookkinds=0,id_num=0,re_num=0;
int counter=0;
char name_r[100]={0};
int permission=0;
char account_now[20]={0};
```

2.8 定义 count()函数——读取文件中的书籍信息写入结构体数组

函数开始时，将全局变量 bookkinds 设置为 0，用于后续统计书籍总数。使用 fopen 函数以读写模式 ("r+") 打开名为 booklist.txt 的文件。这个文件包含了所有书籍的详细信息。通过循环，函数从文件中读取书籍信息。这里使用 fscanf 函数从文件中读取书名、作者、出版社等信息到结构体数组 book 的对应位置。在读取书籍信息的同时，函数检查每本书的馆藏数 (num_z)。如果馆藏数不为 0，说明这本书是有效的，因此将 bookkinds 加 1。完成文件读取和书籍数量统计后，使用 fclose 函数关闭文件。

```
void count() //统计书本数量,并读取文件中的书籍信息写入结构体数组
{
    bookkinds=0;
    fp_book=fopen("booklist.txt","r+");
    for(int i=0;i<N;i++)
    {
        fscanf(fp_book,"%s%s%s%s%d%d%s",book[i].name,book[i].author,book[i].publisher,book[i].ISBN,&book[i].num_z,&book[i].num_x,book[i].date);
        if(book[i].num_z!=0)
        {
            bookkinds++;
        }
    }
    fclose(fp_book);
}
```

2.9 定义 id_read()函数——读取文件中的账户信息写入结构体数组

函数开始时，将全局变量 id_num 设置为 0，用于后续统计用户账号的总数。使用 fopen 函数以读写模式 ("r+") 打开名为 idlist.txt 的文件。这个文件包含了系统中所有用户账号的详细信息，包括账号名、密码和权限。通过循环，函数从文件中读取用户账号信息。这里使用 fscanf 函数从文件中读取并写入到结构体数组 id 的对应位置。在读取用户账号信息的同时，函数检查每个账号的权限等级 (perm)。如果权限等级为 0 或 1（表示学生用户或管理员），说明这个账

号是有效的，然后将 id_num 加 1。完成文件读取和用户账号数量统计后，使用 fclose 函数关闭文件。

```
void id_read()           //统计账号数量,并读取文件中的书籍信息写入结构体数组
{
    id_num=0;
    fp_id=fopen("idlist.txt","r+");
    for(int i=0;i<Nid;i++)
    {
        fscanf(fp_id,"%s%s%d",id[i].account,id[i].password,&id[i].perm);
        if(id[i].perm==0||id[i].perm==1)
        {
            id_num++;
        }
    }
    fclose(fp_id);
}
```

2.10 定义 re_read()函数——读取文件中的借书信息写入结构体数组

函数开始时，将全局变量 re_num 设置为 0，用于后续统计需要归还的书籍总数。使用 fopen 函数以读写模式（"r+"）打开名为 returnlist.txt 的文件。这个文件包含了系统中所有待归还书籍的详细信息，通常包括书名和所属用户账号。通过循环，函数试图从文件中读取每一条信息。这里使用 fscanf 函数从文件中读取每条记录的各个字段（如书名和所属用户账号）到结构体数组 rebook 的对应位置。在读取还书记录的同时，函数检查每条记录的书名字段（retname）。如果书名不为空（表示这是一条有效的还书记录），就将 re_num 加 1。完成文件读取和还书记录数量统计后，使用 fclose 函数关闭文件。

```
void re_read()           //统计借书种数,并读取文件中的借书信息写入结构体数组
{
    re_num=0;
    fp_re=fopen("returnlist.txt","r+");
    for(int i=0;i<Nre;i++)
    {
        fscanf(fp_re,"%s%s",rebook[i].retname,rebook[i].belong);
        if(strlen(rebook[i].retname)!=0)
        {
            re_num++;
        }
    }
    fclose(fp_re);
}
```

2.11 定义 file_w ()函数——将书籍结构体的信息写入文件

使用 fopen 函数以写入模式（"w"）打开名为 booklist.txt 的文件。如果该文件已存在，此操作将清空文件内容；如果文件不存在，则会创建一个新文件。函数通过循环遍历结构体数组 book，该数组包含了系统中所有书籍的详细信息。

对于数组中每本馆藏数不为 0 的书籍（即有效书籍），函数使用 `fprintf` 函数将书籍的各个字段（如书名、作者、出版社等）写入 `booklist.txt` 文件。每写入一本书的信息后，都会添加一个换行符，分隔不同的书籍记录。完成所有书籍信息的写入后，使用 `fclose` 函数关闭文件。

```
void file_w() //将书籍结构体的信息写入文件
{
    fp_book=fopen("booklist.txt","w");
    for(int i=0;i<bookkinds+1;i++)
    {
        if(book[i].num_z!=0)
        {
            fprintf(fp_book,"%s %s %s %s %d %d %s\n",book[i].name,book[i].author,book[i].publisher,book[i].ISBN,book[i].num_z,book[i].num_x,book[i].date);
        }
    }
    fclose(fp_book);
}
```

2.12 定义 `clear()` 函数——整理空书籍

函数通过循环遍历结构体数组 `book`，该数组包含了系统中所有书籍的详细信息。对于数组中的每一本书籍，函数检查其库存数（`num_z` 字段）。如果库存数为 0，表示这本书已经没有馆藏，因此可以被视为无效书籍。如果当前书籍的库存数为 0，函数会将数组中该位置及之后的所有书籍记录向前移动一位，以覆盖掉无效书籍的记录。由于清除了无效书籍记录，因此书籍的总数

（`bookkinds`）也需要相应减少。最后，函数调用 [file_w 函数](#) 将更新后的书籍结构体数组（已清除无效记录）写入到 `booklist.txt` 文件中。

```
void clear() //整理空书籍
{
    for(int i=0;i<N;i++)
    {
        if(book[i].num_z==0)
        {
            for(int j=i;j<N-1;j++)
            {
                book[j]=book[j+1];
            }
        }
    }
    file_w();
}
```

2.13 定义 `list_print()` 函数——输出书的列表

函数首先调用 `count` 函数来读取书籍文件 `booklist.txt`，并统计系统中书籍的总数。接着，函数通过循环遍历结构体数组 `book` 依次处理每一本有效的书籍。

对于数组中的每一本书籍，函数使用 `printf` 函数打印出书籍的序号（通过数组索引计算得出）和书名。等待用户输入：打印完所有书籍信息后，函数会提示用户按任意键返回主界面。

```
void list_print() //不清除显示内容的情况下输出书的列表
{
    int i;
    count();
    for(i=0;i<bookkinds;i++)
    {
        printf("\t\t\t\t\t| %d----- 《%s》 \n",i+1,book[i].name);
    }
}
```

2.14 定义 `refile_w()` 函数——将还书结构体的信息写入文件

使用 `fopen` 函数以写入模式（"w"）打开名为 `returnlist.txt` 的文件。如果该文件已存在，此操作将清空文件内容；如果文件不存在，则会创建一个新文件。函数通过循环遍历结构体数组 `rebook`，该数组包含了系统中所有用户的还书记录。对于数组中每条有效的还书记录（即书名不为空），函数使用 `fprintf` 函数将记录的各个字段（如书名和所属用户账号）写入 `returnlist.txt` 文件。每写入一条还书记录后，都会添加一个换行符，完成所有还书信息的写入后，使用 `fclose` 函数关闭文件。

```
void refile_w() //将还书结构体的信息写入文件
{
    fp_re=fopen("returnlist.txt","w");
    for(int i=0;i<Nre;i++)
    {
        if(strlen(rebook[i].retname)!=0)
        {
            fprintf(fp_re,"%s %s\n",rebook[i].retname,rebook[i].belong);
        }
    }
    fclose(fp_re);
}
```

2.15 定义 `sign()` 函数——登录页

函数首先使用系统命令 `system("cls")` 清除屏幕内容，并显示一个登录界面，包括欢迎信息、登录选项以及退出选项。提示用户输入账号和密码，并使用 `scanf` 函数从标准输入读取用户的输入。通过调用 [id_read 函数](#)，从 `idlist.txt` 文件中读取所有用户账号的信息，并存储到结构体数组 `id` 中。遍历结构体数组 `id`，检查用户输入的账号和密码是否与文件中存储的任何一个账号信息匹配。如果找到匹配的账号，则记录其权限等级（管理员或学生）。如果用户输入的账号和密码正确，则根据用户的权限等级显示相应的欢迎信息，并调用 [menu 函数](#)

```
void sign() //登录页
{
    system("cls");
    int n,a=0;
    char geta[20],getp[20];
    char zhanghu[2][10]={"管理员","学生"};
    id_read();
    printf("\n\n");
    printf("\t\t\t\t*****\n");
    printf("\t\t\t\t*          @@@@   @@@@   @   @   @@@@@@*\n");
    printf("\t\t\t\t*          @      @   @@   @   @       *\n");
    printf("\t\t\t\t*          @      @     @   @   @       *\n");
    printf("\t\t\t\t*          @  @@@   @     @   @   @       *\n");
    printf("\t\t\t\t*          @    @   @     @@   @   @       *\n");
    printf("\t\t\t\t*          @@@@   @@@@   @@@@   @\n");
    printf("\t\t\t\t*****\n");
    printf("\t\t\t\t*          -----欢迎使用图书管理系统-----*\n");
    printf("\t\t\t\t* \t\t\t1 : ---- 账号登录\t\t\t\t*\n");
    printf("\t\t\t\t* \t\t\t2 : ---- 游客系统\t\t\t\t*\n");
    printf("\t\t\t\t* \t\t\t3 : ---- 退出系统\t\t\t\t*\n");
    printf("\t\t\t\t*****\n");
    printf("\t\t\t\t>>>输入您需要使用的功能，并按回车键：");
    scanf("%d",&n);
    getchar();
    if(n==1)
    {
        printf("\n\t\t\t\t请输入帐号：");
        scanf("%s",geta);
        printf("\t\t\t\t请输入密码：");
        scanf("%s",getp);
        for(int i=0;i<id_num;i++)
        {
            if(strcmp(geta,id[i].account)==0)
```

```

        {
            a=1;
            if(strcmp(getp,id[i].password)==0)
            {
                permission=id[i].perm;
                strcpy(account_now,id[i].account);
                printf("\t\t\t\t%s 登录成功 ! \n\t\t\t\t\t 输入任意字符
进入菜单 : ",zhanghu[id[i].perm]);
                scanf("%s");
                menu();
            }
            else
            {
                printf("\t\t\t\t\t>>>密码错误,请重新登录 ! \n\t\t\t\t\t>>>
输入任意字符返回登录 : ");
                scanf("%s");
                sign();
            }
        }
    }
    if(a==0)
    {
        printf("\t\t\t\t\t>>>账号不存在,请重新登录,或咨询图书管理员开通账
号 ! \n\t\t\t\t\t>>>输入任意字符返回登录 : ");
        scanf("%s");
        sign();
    }
}
else if(n==2)
{
    printf("\n\n\n\t\t\t\t\t 确定使用游客登录,按 y 确认,输入其他字符回到登
录页面 : ");
    scanf("%s",&a);
    if(a=='Y' || a=='y')
    {
        menu_yk();
    }
    else
    {
        sign();
    }
}
else if(n==3)
{
    exit(0);
}
else
{

```

```

        sign();
    }
}

```

2.16 定义 menu()函数——菜单页

函数首先清除屏幕内容，并显示一个主菜单界面。这个界面通常包括欢迎信息、查看书籍清单、借阅书籍、归还书籍等功能选项以及退出选项。提示用户输入要执行的功能编号，并使用 `scanf` 函数从标准输入读取用户的输入。根据用户输入的功能编号，执行相应的功能函数。例如，如果用户输入 1，则调用 `list` 函数显示书籍清单；输入 2，则调用 `borrow` 函数处理借书操作；以此类推。如果用户输入的功能编号无效（如超出范围或不存在对应功能），则通常重新显示主菜单并提示用户重新输入。如果用户执行了登录或退出登录操作（通过选择特定功能编号），则更新用户的登录状态。例如，如果用户选择了退出登录，则调用 `sign` 函数处理退出登录操作。函数设计为一个循环，以便在用户完成一个功能操作后能够返回到主菜单，并继续选择其他功能。这种循环结构确保了用户可以在系统中自由导航，并执行多个连续的操作。

```

void menu()                                //菜单页
{
    system("cls");
    int x;
    printf("\n\n");
    printf("\t\t\t\t\t*****\n");
    printf("\t\t\t\t\t@@@@@  @@@@@  @  @  @@@@@@@\n");
    printf("\t\t\t\t\t@      @  @@  @  @      \n");
    printf("\t\t\t\t\t@      @  @  @  @      \n");
    printf("\t\t\t\t\t@  @@@  @  @  @  @      \n");
    printf("\t\t\t\t\t@  @  @  @@  @  @      \n");
    printf("\t\t\t\t\t@@@@@  @@@@@  @@@@  @\n");
    printf("\t\t\t\t\t*****\n");
    printf("\t\t\t\t\t1 : ---- 查看书籍清\n");
    printf("\t\t\t\t\t*\n");
    printf("\t\t\t\t\t2 : ---- 借阅书\n");
    printf("\t\t\t\t\t*\n");
    printf("\t\t\t\t\t3 : ---- 归还书\n");
    printf("\t\t\t\t\t*\n");
    if(permission==0)
    {

```

[illegible]

```

        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t*****\n");
        printf("\t\t\t\t 非管理员账号,无法修改 ! \n\t\t\t\t 如有需求请
        咨询图书管理员 ! \n\t\t\t\t 按任意键返回菜单 : ");
        scanf("%s");
        menu();
    }
    if(permission==0)
    {
        update();
    }
}
else if(x==5)
{
    sign();
}
else
{
    menu();
}
}
}

```

2.17 定义 menu_yk()函数——游客菜单页

函数首先清除屏幕内容，并显示一个特别为游客设计的菜单界面。这个界面与主菜单类似，但通常只包含一些无需身份验证即可使用的功能，如查看书籍清单。提示游客输入要执行的功能编号，并使用 scanf 函数从标准输入读取游客的输入。根据游客输入的功能编号，执行相应的功能函数。由于游客模式限制了可访问的功能，因此可执行的功能通常只有查看书籍清单等。如果游客输入的功能编号无效或超出了游客可访问的范围，则重新显示游客模式菜单并提示游客重新输入。如果游客选择返回登录界面或退出程序，则分别调用相应的函数来处理。与主菜单函数类似，menu_yk 函数也通常设计为一个循环，以便游客在完成一个功能操作后能够返回到游客模式菜单，并继续选择其他功能。

```

void menu_yk()          //用于游客模式的菜单
{
    system("cls");
    int x;

```

```

printf("\n\n");
printf("\t\t\t\t\t*****\n");
printf("\t\t\t\t\t      @@@@@@  @@@@@@  @  @  @@@@@@@@
*\n");
printf("\t\t\t\t\t      @      @  @@  @  @      @
*\n");
printf("\t\t\t\t\t      @      @  @  @  @  @      @
*\n");
printf("\t\t\t\t\t      @  @@@  @      @  @  @      @
*\n");
printf("\t\t\t\t\t      @  @  @  @@  @  @      @
*\n");
printf("\t\t\t\t\t      @@@@@@  @@@@@@  @@@@@  @
*\n");
printf("\t\t\t\t\t*****\n");
printf("\t\t\t\t\t      游客账号登录成
功 !          *\n");
printf("\t\t\t\t\t      仅可查看书籍清
单          *\n");
printf("\t\t\t\t\t      1 : ---- 查看书籍清
单          *\n");
printf("\t\t\t\t\t      2 : ---- 回到登录界
面          *\n");
printf("\t\t\t\t\t      3 : ---- 退出程
序          *\n");
printf("\t\t\t\t\t*****\n");
printf("\t\t\t\t\t>>>请输入您要使用的功能 : ");
scanf("%d",&x);
getchar();
if(x==1)
{
    list_yk();
}
if(x==2)
{
    sign();
}
else if(x==3)
{
    exit(0);
}
else
{
    menu_yk();
}

```

}

2.18 定义 list() 函数——书籍清单页

使用系统命令 `system("cls")`清除屏幕上的内容，以便在一个干净的环境下显示书籍列表。调用 `count` 函数来读取书籍文件 `booklist.txt`，并统计系统中书籍的总数。这个步骤同时也确保了结构体数组 `book` 中包含了所有有效的书籍信息。通过循环遍历结构体数组 `book`，对每一本有效的书籍（即馆藏数不为 0）进行处理。对于数组中的每一本书籍，使用 `printf` 函数打印出书籍的序号（通过数组索引计算得出）和书名。在打印完所有书籍信息后，函数会提示用户按任意键，通过 `menu()` 函数返回主界面。

```
void list() //书籍清单页
{
    int i;
    system("cls");
    count();
    printf("\n\n");
    printf("\t\t\t\t\t*****\n");
    printf("\t\t\t\t\t      @@@@@@   @@@@@@   @       @       @@@@@@@@@\n");
    printf("\t\t\t\t\t      @         @     @@    @       @           @\n");
    printf("\t\t\t\t\t      @         @         @    @       @           @\n");
    printf("\t\t\t\t\t      @  @@@    @        @    @       @           @\n");
    printf("\t\t\t\t\t      @      @  @      @@@    @       @           @\n");
    printf("\t\t\t\t\t      @@@@@@   @@@@@@   @@@@@@   @\n");
    printf("\t\t\t\t\t*****\n");
    for(i=0;i<bookkinds;i++)
    {
        printf("\t\t\t\t\t\t\t| %d----- 《%s》 \n",i+1,book[i].name);
    }
    printf("\t\t\t\t\t*****\n");
    printf("\t\t\t\t\t>>>按任意键回到主界面 : ");
    scanf("%s");
    menu();
}
```

2.19 定义 list_yk()函数——游客书籍清单页

使用系统命令 `system("cls")` 清除屏幕上的内容，提供一个干净的界面来查看书籍列表。调用 `count` 函数来读取书籍文件 `booklist.txt`，并统计系统中书籍的总

数通过循环遍历结构体数组 `book`，对每一本有效的书籍（即馆藏数不为 0）进行处理。对于数组中的每一本书籍，使用 `printf` 函数打印出书籍的序号（通过数组索引计算得出）和书名。在打印完所有书籍信息后，函数会提示游客按任意键返回到游客模式的主菜单（即 `menu_yk()` 函数）。

```
void list_yk() //游客模式的书籍清单页
{
    int i;
    system("cls");
    count();
    printf("\n\n");
    printf("\t\t\t\t*****\n");
    printf("\t\t\t\t      @@@@@@   @@@@@@   @       @       @@@@@@@@@@  
*\n");
    printf("\t\t\t\t      @           @     @@   @       @           @  
*\n");
    printf("\t\t\t\t      @           @       @   @       @       @  
*\n");
    printf("\t\t\t\t      @  @@@   @       @   @       @       @  
*\n");
    printf("\t\t\t\t      @     @   @       @@   @       @       @  
*\n");
    printf("\t\t\t\t      @@@@@@   @@@@@@   @@@@@@   @  
*\n");
    printf("\t\t\t\t*****\n");
    for(i=0;i<bookkinds;i++)
    {
        printf("\t\t\t\t\t\t\t| %d----- 《%s》 \n",i+1,book[i].name);
    }
    printf("\t\t\t\t*****\n");
    printf("\t\t\t\t>>>按任意键回到主界面 : ");
    scanf("%s");
    menu_yk();
}
```

2.20 定义 borrow()函数——借书

使用系统命令 `system("cls")` 清除屏幕上的内容，以便在一个干净的环境下显示借书界面。调用 [count 函数](#) 来读取书籍文件 `booklist.txt`，并统计系统中书籍的总数。提示用户输入要借出的书籍名称（支持模糊查询），并使用 `scanf` 函数从标准输入读取用户输入的书籍名称。根据用户输入的书籍名称，通过循环遍历结构体数组 `book` 中的每一本书籍，进行模糊查询匹配。查询时，不仅考虑书名，还可能在一定条件下考虑作者和出版社信息。

【模糊查询】的实现：根据用户输入的查询内容长度，对结构体数组 `book` 中的每一本书籍的书名、作者和出版社信息进行预处理。通过创建一个二维字符数组 `b` 来存储每个书名、作者和出版社的所有可能子串，然后使用嵌套的循环结构遍历结构体数组 `book` 中的每一本书籍以及它们的所有可能子串。对于每个子串，使用 `strcmp` 函数与用户输入的查询内容进行比较，如果匹配成功，则将该书籍的信息标记为已查询。

```
for(int i=0;i<bookkinds;i++)                                //模糊查询
{
    for(int k=0;k<strlen(book[i].author);k++)
    {
        if(strcmp(name_r,b[i][k])==0&&book[i].f_query!=1)
        {
            printf("\t %d-----\n",i+1,book[i].name,book[i].author,book[i].publisher,book[i].num_z,book[i].num_x);
            book[i].f_query=1;
            zt=1;
            break;
        }
    }
}
```

将模糊查询匹配到的书籍信息展示给用户，包括书籍序号、书名、作者、出版社、馆藏数和可借数等。提示用户输入书籍序号来确认借书操作，并允许用户核对书籍信息后选择是否借书（通过输入'Y'或'y'确认，输入其他字符取消或返回）。如果用户确认借书，并且书籍的可借数大于 0，则更新书籍的可借数（减 1），将借书记录添加到还书结构体数组 `rebook` 中，并将所有书籍信息和还书记录写入文件（通过调用 `file_w` 和 `refile_w` 函数）。同时，根据用户的选择，决定是否继续借书操作或返回主菜单。如果在处理过程中发现任何错误（如用户输入的书籍序号无效、书籍库存不足等），则向用户显示相应的错误提示，并允许用户选择继续借书、查看书籍清单或返回主菜单。

因源代码很长，故不在此处展示，双击右侧文件打开源代码：

2.21 定义 `ret()` 函数——还书

使用系统命令 `system("cls")` 清除屏幕上的内容，为用户提供干净的界面来执行还书操作。调用 `re_read()` 函数读取还书记录文件 `returnlist.txt`，将内容加载到结构体数组 `rebook` 中。同时，将所有还书记录的 `r_query` 字段初始化为 0，

表示这些记录还未被查询。遍历结构体数组 `rebook`，查找属于当前登录用户的还书记录（通过比较 `belong` 字段与当前登录用户的账号）。对于属于当前用户的记录，将其 `r_query` 字段设为 1，表示这些记录已被查询并可用于后续处理。将过滤后属于当前用户的还书记录展示给用户，包括书籍名称和编号，确认要归还的书籍信息。提示用户输入要归还的书籍编号，并使用 `scanf` 函数从标准输入读取用户输入的编号。检查用户输入的书籍编号是否存在于当前用户的还书记录中。如果无编号，则提示用户重新输入。如果书籍编号有效，则在结构体数组 `book` 中查找对应的书籍记录，并更新其可借数（加 1），表示书籍已归还。同时，从结构体数组 `rebook` 中删除该还书记录，确保数据的一致性。调用文件操作函数（如 `file_w` 和 `refile_w`）将更新后的书籍信息和还书记录写入文件，在完成还书操作后，提示用户是否继续归还其他书籍或返回主菜单。根据用户的选择，执行相应的操作。

因源代码很长，故不在此处展示，双击右侧文件打开源代码：

2.22 定义 `update()` 函数——修改图书信息

使用系统命令 `system("cls")` 清除屏幕上的内容，在屏幕上显示一系列更新选项，包括添加书籍信息、修改书籍信息、删除书籍信息和返回上一级菜单。提示用户输入要执行的操作选项，并使用 `scanf` 函数从标准输入读取用户的选择。提示用户输入新书籍的各项信息，如书名、作者、出版社、ISBN 号、总库存量和现有库存量等。将新书籍信息添加到结构体数组 `book` 的末尾，并更新全局变量 `bookkinds` 以反映新的书籍总数。将更新后的书籍信息写入书籍文件 `booklist.txt`，确保数据的持久性。

如果用户选择修改，提示用户输入要修改的书籍序号。根据用户输入的序号，在结构体数组 `book` 中找到对应的书籍记录。提示用户输入新的书籍信息，并更新数组中的相应字段。调用 `clear()` 函数整理空书籍记录（如果需要的话），并重新写入书籍文件。

如果用户选择删除，提示用户输入要删除的书籍序号。根据用户输入的序号，在结构体数组 `book` 中将对应书籍的总库存量设为 0（表示该书籍不再馆藏）。调用 `clear()` 函数整理空书籍记录，并从书籍文件中删除相应的条目。

如果用户选择返回，调用 `menu()` 函数。

错误处理：在处理用户输入和执行更新操作时，检查可能出现的错误情况（如输入无效序号、输入格式错误等），并向用户显示相应的错误提示。

因源代码很长，故不在此处展示，双击右侧文件打开源代码：

2.23 主函数

通过主函数设置画面背景为黑色，然后将借出书的信息和未借出书的信息写入结构体中，进入登录界面 `【sign()】`，在登录界面可进入主菜单 `【menu()】`

与游客菜单【[menu_yk\(\)](#)】，主菜单可进入查看书籍清单【[list\(\)](#)】、借阅书籍【[borrow\(\)](#)】、归还书籍【[ret\(\)](#)】、修改图书信息【[update\(\)](#)】等函数，也可返回登录界面【[sign\(\)](#)】。游客菜单可进入游客书籍清单查询页【[list_yk\(\)](#)】与返回登录界面【[sign\(\)](#)】。

```
int main()
{
    system("color 00");
    re_read();
    count();
    sign();
}
```

3 使用说明

3.1 启动系统

将 library.exe 与 booklist.txt、idlist.txt、returnlist.txt 置于同一文件下，双击 library.exe 启动图书馆管理系统。

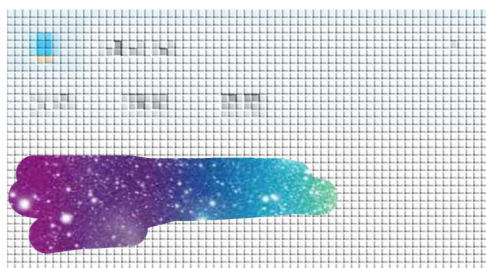
3.2 输入

1) 书库清单 (booklist.txt)

以书名、作者、出版社、ISBN 号、馆藏数、可借数、修改日期的顺序存放的书籍信息，单本书各项之间以空格分隔，不同书籍换行分隔。

2) 账户清单 (idlist.txt)

以账户、密码、权限组的顺序存放的账户信息，单个账号各项以空格分隔，不同账号换行分隔。



3) 借阅记录 (returnlist.txt)

以书名、借阅者账户顺序存放的借阅记录，单条记录以空格分隔，不同记录换行分隔。

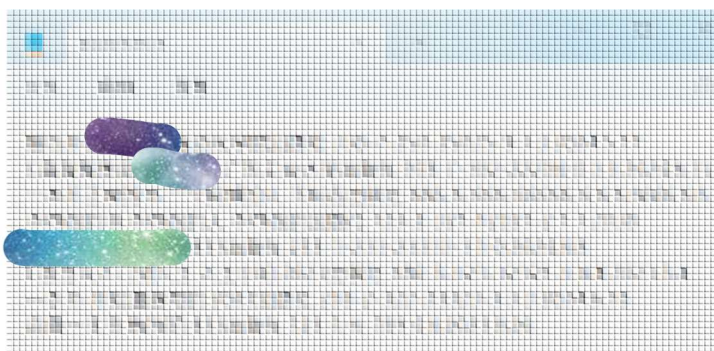
4) 用户命令行输入

用户按屏幕提示所输入的数据，其输入规则符合提示要求。

3.3 输出

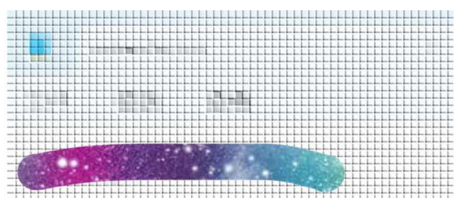
1) 修改后的书库清单 (booklist.txt)

在程序中修改后的以书名、作者、出版社、ISBN 号、馆藏数、可借数、修改日期的顺序存放的书籍信息，单本书各项之间以空格分隔，不同书籍换行分隔。







2) 借阅书籍产生的借阅记录 (returnlist.txt)

在借阅操作完成后产生的以书名、借阅者账户顺序存放的借阅记录，单条记录以空格分隔，不同记录换行分隔。



3) 命令行界面输出

用于显示程序信息及提示用户操作的信息。

(由于大作业的提交要求里只允许提交源代码和报告，为了方便测试，只在代码里定义了两本书——《》和《C 语言程序设计》，以及两个账户：管理员账号密码 123456，借阅了自传》；学生账号借阅了《C 语言程序设计》。)

4 运行测试

1) 进入登录界面，输入账号，自动识别管理员、学生和游客身份并登录：



- 2) 以管理员（学生）身份登录后，进一步选择：

```
*****
*          1 : ---- 查看书籍清单          *
*          2 : ---- 借阅书籍              *
*          3 : ---- 归还书籍              *
*          4 : ---- 修改图书信息          *
*          5 : ---- 账号登出              *
*****
>>>请输入您要使用的功能：|
```

- 3) 输入 1 查看书籍清单:

	「痛名ノ名」
ノ	「益名解理分ノ」
シ	「名ノニ持名也」
ン	「ノ解名也」
ン	「安難名也」
カ	「益名分ノ御ノ」
	「名也」
カ	「少解名也」

- 4) 输入 2 进行书籍借阅:

[illegible]

进一步确认所需书籍，输入书籍编号确定书籍是否有所剩余，是否借阅：

[illegible]

若书籍数目不够或不想借书, 则将进一步选择继续借书或者退出。

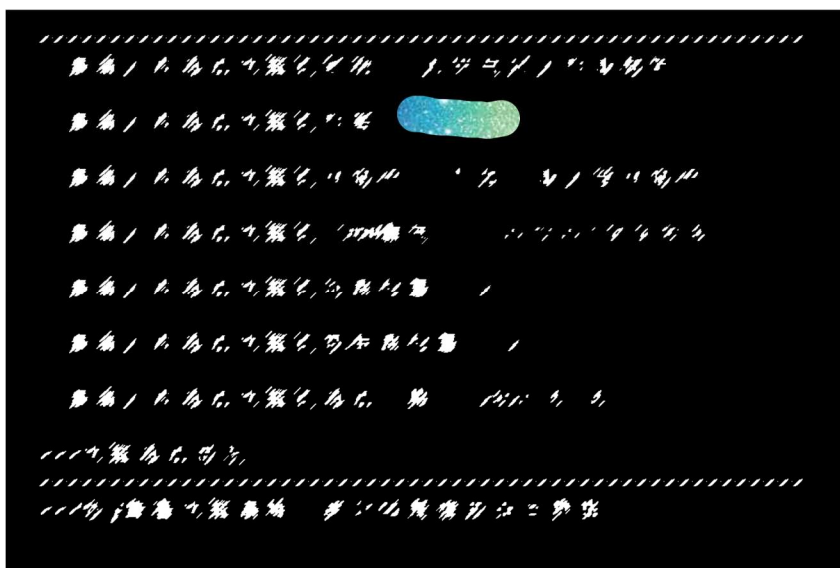
- 5) 输入 3 进行归还书籍:

~~~~~  
~~~~~  
~~~~~  
~~~~~  
~~~~~  
~~~~~

- 6) 输入 4 进行修改书籍:

```
*****
*                1-----添加一本书籍信息                *
*                2-----修改一本书籍信息                *
*                3-----删除一本书籍信息                *
*                4-----返回上一级                        *
*****
>>>请输入您要使用的功能：|
```

7) 输入相关书籍信息以添加:



输入书籍编号进行修改, 具体操作同上

8) 输入 5 退出账号

5 问题及解决办法、心得体会

万事开头难, 在开始写大程序的时候, 我完全不知道怎么开始、从哪开始, 完全不知道一个好的程序应该具备怎样的条件。百般无奈下我请教了本专业的师兄和计算机学院的同学, 在他们的建议下, 我选择了制作图书管理系统, 并且在他们的指导下, 我在网络上找到了一些诸如备忘录、银行系统的程序代码, 也找到类似的图书管理系统代码。我努力学习并模仿他们分析和解决问题的方法, 对要制作的图书管理系统有了大致的构造。

其次就是在编写函数时, 因为整个程序需要定义较多的变量名, 我需要确保我输入的变量名是正确的, 尤其是数组、结构体这些较长的名称, 十分耗费时间和耐心。但同时, 我也体会到变量名要有一定意义的重要性。然而, 随着思考的加深和功能的完善, 我渐渐意识到系统中每个功能、每个操作类与数据类之间都密切相关。没有任何数据或功能可以独立于其他成员存在, 每次修改系统代码都会影响整个系统。这也是为什么我单独完成整个系统的编写的原因。随着对系统的反复修改, 我深刻体会到, 一个系统的数据结构比功能更为重要。如果不理解数据类与操作类之间的联系, 不清楚它们应该如何配合, 编写的代码就会混乱不堪, 功能实现也会面临各种问题。相反, 理解它们之间的内在联系会让思路更为清晰, 有利于功能的实现。

尽管图书管理系统仍存在一些缺陷, 代码可能不够简化, 但算是告一段落了。回顾这个过程, 我认为数据结构的理解与处理才是关键, 今后, 我会不断巩固我的程序设计能力, 做到灵活运用和融会贯通。