## 使用说明：

1. 选择操作的时候请输入整型操作数，不然会判定为无效输入
2. 书名、作者、ID等要输入字符串的请在16个字符以内，系统只会识别你输入的前16个字符，输入30个字符以上系统会出错。
3. 字符串输入的支持汉字输入，8个汉字以内是正确输入
4. 输入借阅日期的时候虽然可以超出范围，但是超出99天之后系统会默认为你要借阅99天，另外，因为系统无法将借阅时间逐日减少，因此通过借阅者所填写的借阅期限去判断到期时间。
5. 本系统没有连接数据库，因此关闭程序之后你之前的操作都会被清空，数据又会初始化，能力不足，敬请谅解

    6.若有不明白之处，请将问题指出来，发邮箱到1642801201@qq.com.

7.感谢你的使用

# 项目简介

这是一个被限制在可恶的黑框内的图书管理系统，没有漂亮的交互界面，也没有连接数据库，黑框打开->数据生成，黑框关闭->数据消失，是不是很可恶呢！

这个系统的功能有：

1. 查看书：即遍历书库中所有的图书，查看各本书的书名、作者、库存、现存、是否被预约等信息。
2. 增加书：增加一本新的书进入到书库中，其中用户需要输入书名和该书的作者，如果有同名的书已经在书库中，则该种书数量增加1，否则定义该种书的数量为1。
3. 注销书：若一本书已经没有了保留价值，输入该书的书名便可将该书从书库中注销。
4. 借阅书：若某本书在书库中并且现存量大于零，便可借出。借阅者需要输入自己的借阅证号和借阅时间（1天-99天）
5. 预约取书：当某本书在书库中并且现存量大于零，便可预约什么时候取书，借阅者需要输入自己的借阅证号、取书时间，借阅时间
6. 归还书：输入借阅证号和想要归还的书本名字便可以归还书本
7. 查作者：即搜寻该作者的全部书籍，并输出相关的信息
8. 查书：输入某本书的书名，便可以查看该书的全部信息，还可以输出该书的借阅者信息，并计算最早逾期的借阅者证号。

# 总体设计

本程序建立了两条链表，一条存放书籍的相关信息，一条存放借阅者的相关信息。

1. **typedef** **struct** TNode          //该结点储存书籍的相关信息
2. {
3. **struct** TNode \*next;
4. **struct** TNode \*prev;
5. **struct** TNode \*rear;
7. **int**  bookNum;           //书的编号
8. **char** bookName[30];
9. **char** bookAuthor[30];
10. **int**  standingStock;    //现存量
11. **int**  totalStock;      //库存量
12. **int**  able2lend;      //可外借的书籍（部分被预约）
14. }TNode, \*ptrTNode;
15. **typedef** **struct** LNode                //该结点储存借阅者的相关信息
16. {
17. **struct** LNode \*next;
18. **struct** LNode \*prev;
19. **struct** LNode \*rear;
21. **int**  appointDate;            //预约时间期限
22. **int**  limitDate;            //借阅期限
23. **char** bookName[30];
24. **char** ID[30];             //图书证号
26. }LNode, \*ptrLNode;

然后实现逐个功能，下面给出相关的函数名

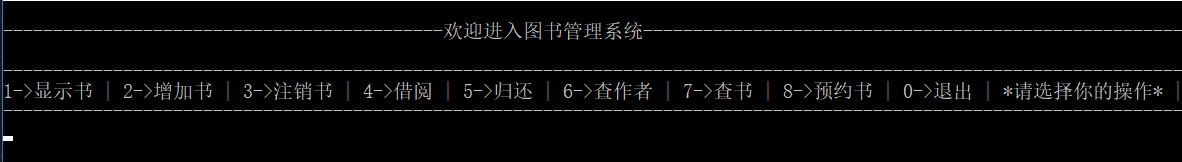
1. //建立书库
2. **void** create1(ptrTNode Q);
4. //建立储存借阅者信息的链表
5. **void** create2(ptrLNode Q);
7. //查看库存的书
8. **void** print(ptrTNode Q);
10. //添加书本的操作
11. **void** addOperate(ptrTNode Q);
13. //添加一本书到书库中
14. **void** addBook(ptrTNode Q,**char** \*bookName,**char** \*bookAuthor);
16. //注销书本
17. **void** logoutBook(ptrTNode Q);
19. //借阅书本
20. **void** lendBook(ptrTNode Q,ptrLNode P);
22. //归还书本
23. **void** returnBook(ptrTNode Q,ptrLNode P);
25. //搜寻某一个作者的全部书籍
26. **void** searchBook2(ptrTNode Q);
28. //搜寻某本书的全部信息
29. **void** searchBook3(ptrTNode Q,ptrLNode P);
31. //预约借书
32. **void** appoint(ptrTNode Q,ptrLNode P);
34. //格式需要，将字符串统一为16个字符
35. **char**\* format(**char**\* Name);
37. //交互界面
38. **void** Interface(**void**);
40. //添加借阅者的信息
41. **void** addimfor(ptrLNode P,**char** \*ID,**int** limitDate,**char** \*bookName,**int** appointDate);
42. //搜寻借阅者所借的书本
43. **void** searchBook1(ptrTNode Q,ptrLNode q);

### 下面给出大致结构图：

本系统显然是采用多文件编程的方式，将不同功能的函数分别建立不同的文件（某些关系密切的除外），这样子整个程序就会显得条例清晰，容易修改维护。

# 详细设计

### 首先用 inerface函数简单设计了一个交互界面



### create1()和create2()函数

通过molloc函数开辟空间，建立两个链表的头结点，将各项参数初始化。其中create1函数中包含初始化函数initBookStock()，作用是首先自动添加5本书进去书库里面，以便后续用户的操作。

### addBook()函数

先让用户输入想要增加的书名和作者，然后将书名与书库中的书名依次比较，判断书名是否有相同的，如果有同名的书已经在书库中，则该种书数量增加1，否则定义该种书的数量为1，并添加一个结点到链表中，储存该种书的相关信息。该函数基本上是用循环和判断实现，若想要了解，请观看代码文件。

### logoutBook()函数

跟上面差不多，用户输入想要注销的书名，然后用循环遍历书库，判断书库中是否有对应的书，若有，就释放该结点的空间，若没有就输出注销失败。

### lendBook()函数

首先让借阅者输入自己要借的书，依然采用循环遍历找出该书，显示出该书的信息，若可以外借则让借阅者输入图书证号以及借阅期限，外借成功就将现存和可外借的数量减1，外借失败就显示出来。这里包含一个addimfor()函数，将借阅者的信息都保存下来

### returnBook()函数

因为功能类似，这个函数跟lendBook函数放在同一个文件里面，实现方法与前面所陈述的一样，用户输入借阅证号和书名便可以归还书本。该函数里面还包含着一个searchBook()函数,这个函数的作用是查找出来用户想要归还的书，因为多次查找，封装起来的话直接使用就很方便。

### searchBook2()函数

该函数的功能是查找某一个

作者的全部书籍，只要以作者为索引，找出相应的指针，便可以得到该作者的其他书籍的全部信息。

### appoint()函数

该函数的功能是预约书本，实现方式与lendBook()类似，只不过是改变的参数不一样，这里包含一个addimfor()函数，作用是把该预约者的相关信息存到链表里面取，以便取用。

### searchBook3()函数

该函数查询某一本书并输出该书的全部信息，这个函数分两部分输出，一是利用书库信息指针ptrTNode Q 找到对应的书 ，然后把信息输出，二是利用借阅者信息的指针ptrLNode P找到对应的书，然后把信息输出，同时里面含有一个排序函数bubbleSort(ptrLNode \*Array)，作用是搜寻最早逾期的书的借阅者编号。

篇幅有限，只能描述主要思路，若要详细了解，请观看源代码，里面也有注释说明。

# 数据结构与数据处理

## 双向链表

本程序用到了双向链表取储存书本的信息和借阅者的信息。

优点有：

1. 动态分配内存，增加书的时候就分配内存，注销书的时候就释放内存，数据量大的时候可以大大减轻内存压力
2. 因为采用双向的，所以可以随意获得所在结点的前一个结点和后一个结点的地址，比单向的更加方便
3. 增删书本很方便
4. 为之后的拓展功能，例如预约，查找某一位作者的全部书籍，查找某一本书的全部信息，为这些功能提供了更好的结构去实现。
5. 随时可以获得某一个结点的全部信息，非常方便

## 结构体数组

在这个程序中我还用到了结构体数组，ptrLNode Array[30]，作用是储存该本书借阅者的的信息，以便后面排序，从而选出最早逾期的借阅者。

## 其他数据处理

1. 运用了一个format(char\* Name)函数，把各种字符串限定在16个，既保证不会错误输入又可以保证格式的美观。
2. 其他储存字符串的，比如一些书名，ID，作者等，都用小数组开辟，在现有的情况下较为方便和省内存

# 程序亮点

1. 运用了一个format(char\* Name)函数，把各种字符串限定在16个，多于十六个的就切除后面的，少于十六个的就空格补齐。既保证不会错误输入又可以保证格式美观，在后面频繁比较的时候还可以正确得出比较结果
2. 在结构体中还定义了一个结构体尾指针，就是这个尾指针始终指向链表尾部，添加书的时候可以很方便地在尾部添加，然后遍历的时候也是直接取用这个尾指针开始遍历便可，然后当链表发生改变（增添或者删减）之后，只需要改变该尾指针的指向便可，不必频繁返回指针值
3. 程序中需要频繁比较，我在比较的时候巧妙地运用了strcmp函数，也就是当两个字符串相等的时候返回值为0作为依据去判断，而不必一个一个去判断或者另外建立一个函数（好像除了运用这个函数之外，字符串并不能直接判断是否相等）
4. 我觉得建立一个结构体数组也是蛮骚的，一来可以方便储存指定的一本书的借阅者，二来排序的时候可以直接排序，而不需要是链表排序这么麻烦，三呢最后取那个最早逾期的借阅者编号也很方便，直接取Array[0]->ID就可以了
5. 对用户输入错误的处理也很巧妙，话不多说，直接上代码
6. **int** limitDate = 0;
7. scanf("%d",&limitDate);
8. **if**(limitDate == 0) {gets(uselessArr);**return**;}
9. **if**(limitDate > 99)  limitDate = 99;

当需要输入整型数据的时候，如果用户输入错误，便会使程序结束，同时，

“gets(uselessArr)”还会吸收掉用户的无效字符，从而不影响下一步的操作。当需要输入字符数据时，看似可以随便输入，实则内部程序会控制你的字符串长度，从而使错误率大大减少。

6.emmm我觉得界面挺简洁漂亮的，嘻嘻。

# 六、程序兼容性说明

能在win10下完美运行，其他暂不清楚。

# 七、总结与体会

我在实现了基本功能之后再完善了一下界面、输入与反馈、注释三个方面，觉得也差不多了，在有限的时间、有限的能力范围内基本上已经发挥了80%。总体看上去其实没什么问题，但我觉得还是可以有完善的。

1. 关于输出输出人机操作对话的日记问题，我自己也有考虑过，也有上网查过如何实现，但给出的结果都是只有单个字符串这样去输出到文件中，我觉得如果每一步操作都要输出一下，这样会非常繁琐和累赘，而且代码也会显得比较乱，因此我并没有尝试去实现，若有更好的方法还请指教
2. 关于图书预约的问题，我昨天请同学帮我测试一下他提到没有书的时候也应该可以预约，预约成功之后当被预约的书被归还，这本书就立即被借给预约那位同学。但是这样就跟我的预约借书时间相互冲突，如果没有书的时候可以预约，那么就变成了有书的时候无法预约了，还有如果多人预约的话还要排序，把预约先后分出来，每当有书归还就先借给那个最早预约的同学，考虑到比较复杂，并且时间有限，因此没有实现。
3. 程序运行的时候加载出来的时间比较长，要好几秒，本来想着做完之后再去看一看能不能解决，也是限于时间未能解决，本人觉得可能是代码里面的时间复杂度和空间复杂度的问题，占用内存多或者运行时间慢而导致加载慢（不过从代码来看我觉得也没有什么大问题，还请读者帮忙支出）

觉得做这个出来是很多基础知识的积累，本人没有用到一些很高端的操作，有的知识实打实的一步一步用循环和判断去实现，想象自己几天就完成了一个简单的图书管理系统，还是蛮高兴的呢。