北京邮电大学

课程设计报告

图书馆管理系统

理学院2017214102班

梁晓奇-2017212437

教师_____ 成绩_____

教师评语：

_____

_____

___年___月___日

课程设计概述

1.1问题描述

在数字化、信息化的今天，图书馆的日常运转需要更加智能而高效的计算机介入。

1.2设计目的及基本要求

目的是完成一个符合当今需求的高效智能图书馆系统：

基础功能：图书购入登记、用户账户注册注销、用户查询、借阅、归还

预约功能：某一图书类目下 N 人预约的排队处理。当有人还书时，优先通知预约排队的 第一人，预约取消时通知下一位。若预约时间超过十天未还书或预约者未在 收到通知后 3 天内借阅被预约书籍，则取消该预约。

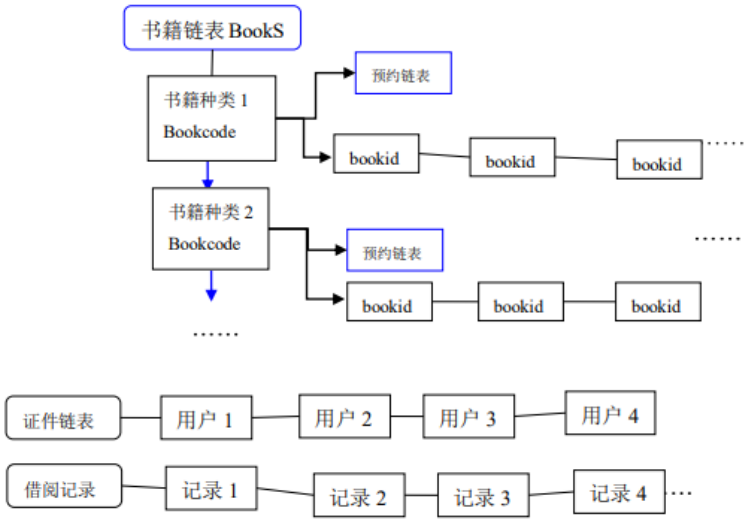
统计功能：某一段时间内，针对人员或书籍进行借阅频率、效率等统计。

1.3设计原理

该系统主要有用户账号和书籍两大数据类。为方便统计和数据调用，我们将借阅记录单独储存。考虑到实际情况中系统使用频率较高的增删改操作及内存分配，书籍、用户账号和借阅记录的存储均采用链表数据结构。

1.4数据结构：

首先，我们以图书类别建立一支链表1，每个节点即为一种图书。在每个节点下接入两支链表：以该种图书下的每本图书为节点的链表 1.1以及该种图书的预约链表 1.2。其次，建立以用户id为key的证件链表2以及所有图书的借阅记录链表3。综上，本设计中共有5个链表来存储数据。证件链表中主要存储每个用户id下的在借书籍数目和 预约书籍数目，借阅记录链表则主要存储所有书籍借出-归还的情况，包括所借书籍的信息和借书者账号、借出与归还的时间和借出时长。本设计中，系统的当前时间即为最 新输入命令的时间，并且加入了时间判断的功能模块来保证闰年时的日期计算。



1.5结构体设计：

本设计主要有5个结构体，进行相关命令操作时主要用到的Key有 Book_code/Book_Id/ Card_code 三个。

```
int NowTime[3] //系统当前时间（全局变量）
int BOOK_ID= 1;
typedef struct Book {
int Buy_time[3];
int Id; //char Code[10];
int Is_Borrowed;//已借出则为 1，否则为0
struct Book * next;//图书 id 的链表，最后一本书指向null
}Book;

typedef struct Card {
char Status;
char Code[10];
char Name[10];
char Department[10];
int Max_Quantity;
int Max_Time;
int Borrow_Num;//目前在借的图书数量，最大数量减去该值便是可借 书籍数量
//int Borrowed_Num;//
int Appoint_Num;//预约书籍数量
struct Card * next;//证件链表
}Card;

typedef struct Appointed_Record {
char Card_Code[10];
char Book_Code[10];
 int Appointed_Time[3];
int Is_Overdue;//该预约记录当前状态，未还0已发通知1
struct Appointed_Record * next;
}Appointed_Record;//预约链表

typedef struct Borrowed_Record {
char Card_Code[10];
```

```
int Book_Id;
char Book_Code[10];
int StartTime[3];
int EndTime[3];
int Borrowed_Days;
struct Borrowed_Record * next;//借阅记录的链表
}Borrowed_Record;
```

课程设计所用到的平台及环境：

Windows:Dev-C++ 5.11
MacOS:Xcode 11.3
Clion 2019.3.1

课程设计内容：

需求分析：

目的是完成一个符合当今需求的高效智能图书馆系统：

基础功能：图书购入登记、用户账户注册注销、用户查询、借阅、归还

预约功能：某一图书类目下 N 人预约的排队处理。当有人还书时，优先通知预约排队的 第一人，预约取消时通知下一位。若预约时间超过十天未还书或预约者未在 收到通知后 3 天内借阅被预约书籍，则取消该预约。

统计功能：某一段时间内，针对人员或书籍进行借阅频率、效率等统计。

概要设计：

基础功能：图书购入登记、用户账户注册注销、用户查询、借阅、归还
预约功能：某一图书类目下 N 人预约的排队处理。当有人还书时，优先通知预约排队的第一人，预约取消时通知下一位。若预约时间超过十天未还书或预约者未在 收到通知后 3 天内借阅被预约书籍，则取消该预约。
统计功能：某一段时间内，针对人员或书籍进行借阅频率、效率等统计。

详细设计：

本设计中设有系统时间变量、日期计算函数及预约处理函数，其中预约处理函数被放置在每个操作命令下，系统外部每进行一次功能/操作则进行一次预约处理。

基础功能：

1.图书购入登记：

RegBooks():若是新的Book_Code则创建新Book节点并相应的添加 Book 节点;若 是已有的 BookCode 则直接找到相应Book节点并添加 Book 节点。实 质为链表节点的增添操作。

2.用户注册、注销：

RegCards：实质即为证件链表的增加操作。成功返回 1，否则返回 0

Destory_Card：实质即为证件链表的删除操作。成功返回 1，否则返回 0

3.查询、借阅&归还

Borrow（某一Book_Id 和用户card_Code）：

先查询该用户所借书是否已超过最大数，再查询该Book_Id是否已被预约，查询该Book_Id是否已借出。若该 Book_Id 无法借阅，此时有 2 种情况：已预约、该Card_code 下无此 Book_Id 的预约记录或预约了未发通知。若该 Book_Id 可借，则借阅成功的同时 增添一条借阅记录，End_Time 设为全 0；Card_Code 中的Borrow_Num+1.若该人是收到通知后来借书，则此时应删除对应的预约记录。

Return（某一Book_Id和用户card_Code）：

以 Book_Id 为 key 遍历借阅记录寻找该 Book_Id 的 End_Time 全 0 的记录并更改相应的 End_Time。随后调用借阅记录中的 Start_Time 来计算借阅时长。若逾期则计算相 关的超期天数，返回须交清的费用。

QueryBookCode（查询该书可借数目）：

检查输入是否合法。成功则返回可借数目，否则返回-1

QueryBookId（该图书借阅情况概要查询）：

在BookS链表下查询该BookId。若最后没有找到对应的记录则说明该书未借出，否则返回记录中的Card_Id。

Overdue_Nums（查找 Card_Code 下当前借的书的逾期记录条数）：

遍历借阅记录时先对比 Card_Code，再对比状态变量，然后再视情况是否对比时间

QueryCard（借阅情况概要查询）：

首先进行输入的合法判断,成功运行返回 1，否则返回 0。调用 Card_Code 中的 4 个变量返回即可

QuerydtCard(借阅情况详细查询)：

首先检查输入的合法性成功的返回 1，否则返回 0。其次，遍历借阅记录查找满足条件 的记录，并且按格式打印；同时计算过期图书的时长

预约功能：

Appoint（某一 Bookcode 和 Cardcode）：

首先调用 Card_Code 中的 Borrow_Num，判断是否可以进行预约；其次检查该Bookcode 是否合法。若合法，则在其节点下的预约链表表尾增添一条记录（节点）。同时，更改cardcode及bookcode节点的Appoint_num。

Appoint_Cancel(人工取消预约)：

在该bookcode下的预约链表中找到对应 cardcode 的预约记录（节点），删除该节点。同时，更改 cardcode 及 bookcode 节点的 Appoint_num。

Appoint_Solve:
首先对当前系统时间进行更新检查。遍历链表 BookS 下每种图书（Bookcode）的预 约链表，对于一条预约链表，我们对第一个节点（记录）进行逾期检查（即预约后没来取的情况）：若逾期，则删除，检查下一节点；若未逾期，则检查是否发了通知。若发 了通知，则检查下一个节点；否则检查当前在库书籍数目是否充足，再判断是否发通知。 该预约处理在其他命令函数下执行。

统计功能：

IsPrime：判断当前年份是否为闰年。

DaysBetween2Date：日期计算函数，计算输入的两个日期之间差了多少天，返回一个整数。

BookTimestat(书籍借阅平均次数统计)：
首先检查统计时间的合法性以及Book_Code 的合法性。其次，以 Time 和 Book_Code 为 key 遍历借阅记录链表，查找归还时间晚于 StartTime 以及借阅时间早于EndTime， 并且Book_Code符合的记录，每找到一个符合要求的记录则计数器+1。成功则返回计数器与书本数量的比值，否则返回-1；

BookEffistat（书籍借阅平均效率统计）：
首先检查统计时间的合法性以及 Book_Code 的合法性。其次，计算该 bookcode 下共有多少本书。然后，以 Time 和 Book_Code 为 key 遍历借阅记录链表，查找借阅时间 在统计区间内并且 Book_Code 符合的记录，用日期计算函数计算出相应的借阅时长，并 将其累加至计数器上。此处，我们对借阅时长跨过 starttime 的情况、恰好在中间的情 况、跨过 endtime 的情况以及同时跨过 starttime 与 endtime 的情况都进行了相应的截 取计算，保证了统计的严谨。最后，对总时间进行处理，计算出借阅平均效率。成功则 返回该值，否则返回-1；

PersonTimestat（人员借阅频率统计）：
首先检查统计时间的合法性以及 Card_Code 的合法性。其次，以 Time 和 Card_Code 为 key 遍历借阅记录链表，查找归还时间晚于 StartTime以及借阅间早于 EndTime，并且借书人为该 Card_Code 的记录，每查到一条则计数器加 1. 成功则返回计数器的值，否则返回-1；

PersonEffistat（人员借阅效率统计）：
首先检查统计时间的合法性以及 Card_Code 的合法性。类似于 BookEffistat，以Time和Card_Code为key遍历借阅记录链表，查找借阅时间在统计区间内并且 Card_Code 符合的记录，用日期计算函数计算出相应的借阅时长，并将其累加至计数器上。此处， 我们对借阅时长跨过 starttime 的情况、恰好在中间的情况、跨过 endtime 的情况以及 同时跨过 starttime 与 endtime 的情况都进行了相应的截取计算，保证了统计的严谨。 最后，对总时间进行处理，计算出借阅效率。成功则返回该值，否则返回-1；

测试

以下是根据指导老师的测试用反复修改的结果及部分更新日志

// Edit by LanXKay on 2019/10/27
// 根据老师的要求更改了时间的流逝方式
// 主要是增加了nowtime以及更改了appoint_solve函数

// Edit by LanXKay on 2019/10/30
// 根据10.29课程设计题目的要求
// 更改bookcode的类型
// book中添加权限变量
// 增加初始化时间函数
// 尝试添加注释

// Edit by LanXKay on 2019/11/5
// 将通过外界输入的int类型变量全部改成了long int
// 通过手写输入函数，解决了scanf中的%d的warning的问题
// 粗略进行了检查，run successed

// Edit by LanXKay on 2019/11/7
// 统计时需要考虑还未结束的借书记录的情况
// 对统计部分的函数做了修改

// Edit by LanXKay on 2019/11/25
// 测试了老师给出的用例，基本没有问题
// 主要测试了1-5以及13

```
OK
OK
OK
OK
OK
0
0
0
OK
OK
OK
OK
OK
0
0
0
OK
0
OK
0
0
0
OK
OK
0
OK
OK
0
OK
0
OK
OK
OK
0
0
0
0
2.00 //
0
0

1.0
0.2
10
3.0
```

课程设计总结：

遇到的一些问题及解决方法：

在调试过程中遇到了许多逻辑上的bug，也就是虽然语法上代码没有bug但是最终的结果与期望的结果有差异。解决方法主要是找到结果不同的测试用例然后对其进行独立测试，也就是通过编译器的debug功能，设置断点等方法通过一步步运行程序找到逻辑问题的根本原因从而进行修改。

对本课的改进意见：

对于本课我无论是从数据结构知识实际运用与合作上都获益匪浅，出于热情与对下届学弟学妹的支持，我们在此提出以下一些建议：
理清学生的知识结构，提高学生课堂效率，激发学生学习兴趣，培养学生分析能力，建议在现有内容上增加新的知识点。

心得体会：

处理该类程序设计问题时，开始的设计环节极为重要，确立数据结构，分析客户需求等，只有这些问题事先想清楚，才有事半功倍的效果。虽然在编码过程中没有太多困难，但是在debug及满足需求方面上着实棘手，对发量也造成了不少真实伤害，不过最后与大家讨论合作后克服诸多困难所获得的成就感也是无与伦比的。特别是在这次设计中我们与老师积极的互动，从老师那得到了许多有用的建议，同时收获了许多难得的实践经验，在这里由衷感谢吴老师的指导。