

**程序设计报告**

实验题目： 图书管理系统模拟

学 期： 2020-2021第二学期

学生姓名： 齐家宝

年 级： 2020级

专 业： 计算机科学与技术

任课老师： 澈力木格

课程名称 程序设计实训

# 程序设计基础综合实验报告

## 实验课题项目：图书管理系统模拟

## 实验目的

1. 通过本次实验，加深对c语言编程的理解
2. 增强分析问题，建立模块，解决问题的能力
3. 巩固c语言基础知识
4. 掌握基础文件处理函数

## 课题要求

1. 问题描述

图书管理信息系统实现图书馆馆藏图书的信息管理与图书借阅。图书管理信息系统包括用户管理、读者管理、图书管理、图书流通功能。

2.整体要求

该系统应具有合理的界面设计，并易于操作；编码风格良好；该系统用控制台程序即可实现；编程语言为C语言，编程环境为VC++ 6.0

## 需求分析

1. 用户登录时输入账号和密码的错误处理：输入错误三次后自动退出
2. 菜单内选项非法输入则提示错误并返回上级菜单
3. 将从键盘输入的相关信息保存到制定文件中，存储结构为数组或链表
4. 每个具体功能都使用自定义函数来进行执行
5. 在控制台程序中显示出具体的菜单界面
6. 可以从文件中提取数据进行浏览
7. 拥有完整的查询系统
8. 拥有完整的账号管理系统
9. 能将文件中的图书或用户信息按照指定关键字进行指定删除
10. 数据的读取与输入输出简洁概要

## 概要设计

用户（用户名、密码、用户类型）

读者（读者号、姓名、单位、联系方式、可借书数、已借书数）

图书（记录号、书号、书名、作者、出版社、藏书量、借出书、指针1、指针2、指针3）

图书查询（name、链头指针、长度）

## 详细设计

1. 结构体设计

//结构体群

struct User//用户

{

char id[100]; //用户名

char password[100]; //密码

char type[100]; //用户类型

struct User\* next;

};

struct Reader //读者

{

char id[100]; //用户名

char company[100]; //单位

char telephone[100]; //联系方式

int can\_borrow; //可借数

int already\_borrow; //已借数

struct Reader\* next;

};

struct Book

{

int num; //序列号

char id[100]; //书号

char type[100]; //书名

char author[100]; //作者

char press[100]; //出版社

int stock; //库存

int borrowed; //借出

int p1;

int p2;

int p3;

struct Book\* next;

};

struct Data

{

int year; //年

int month; //月

int day; //日

};

struct Brrecord

{

char reader\_id[100]; //借书者用户名

char book\_id[100];//书号

struct Data borrow\_data; //借书日期

struct Data return\_data; //还书日期

struct Brrecord\* next;

};

struct Key

{

char keyword[100]; //检索关键字

int head; //头指针

int lengh; //长度

struct Key\* next;

};

1. 功能函数接口设计：

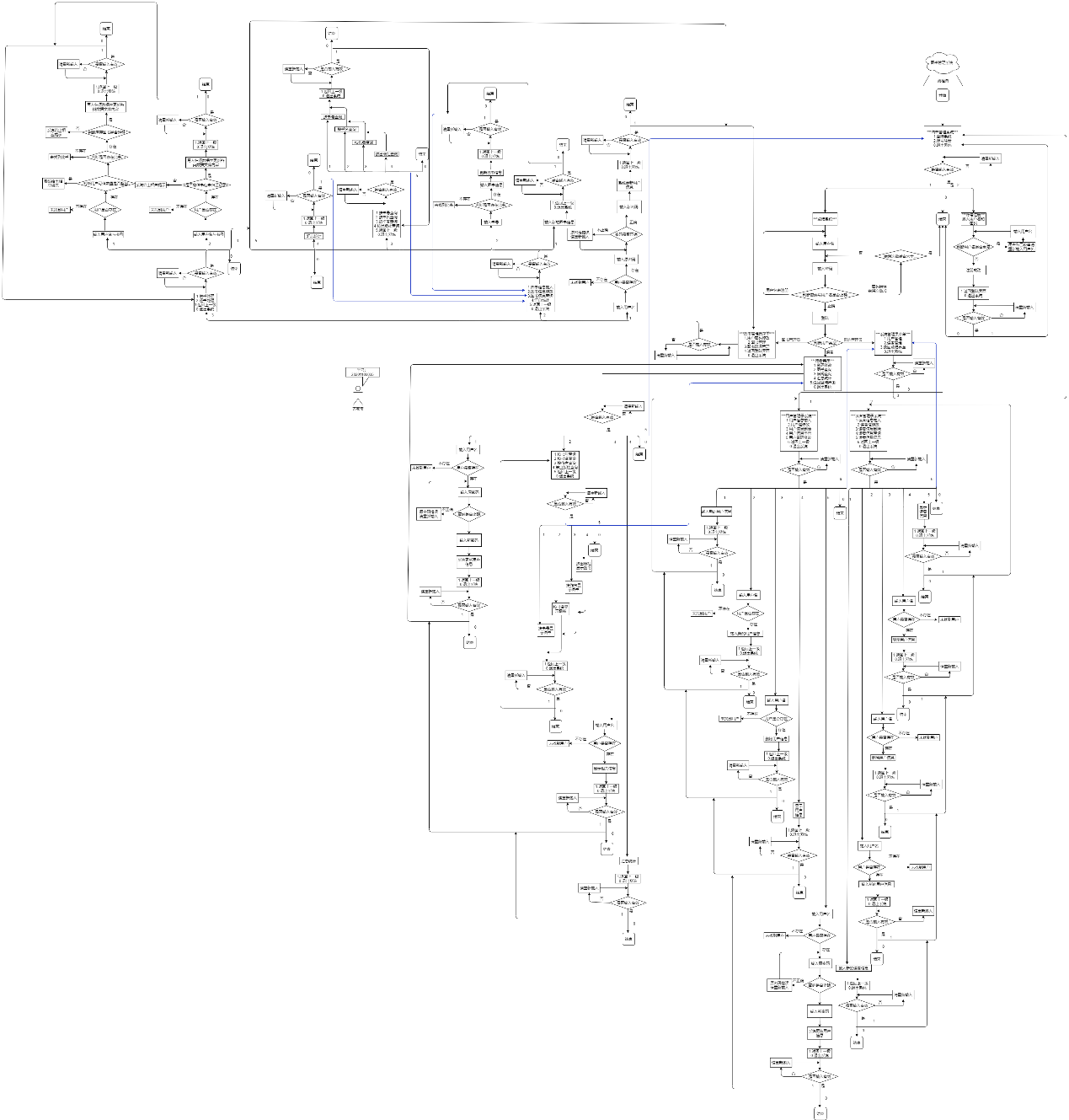
void Encrypted\_input(char\*);//加密输入

void dictionary\_sort\_show(int); //字典排序输出

struct Data current\_system\_time(void); //提取系统当前日期

void clear\_buffer(void); //清空缓冲区

1. 各模块调用流程图



## 源代码：

void Encrypted\_input(char\* str) //加密输入

{

char temp\_password[100];

int i = 0;

while ((temp\_password[i] = \_getch()) != '\r')

{

printf("\*");

i++;

}

temp\_password[i] = '\0';

strcpy(str, temp\_password);

}

void dictionary\_sort\_show(int n)

{

if (1 == n)

{

struct User\* head = NULL, \* p, \* p1, \* p2, \* x, \* t;

p = temp\_storage\_user(head);

p1 = p->next;

p->next = NULL;

while (p1 != NULL)

{

x = p;

p2 = p->next;

while (p2 != NULL && strcmp(p2->id, p1->id) <= 0)

{

x = p2;

p2 = p2->next;

}

t = p1->next;

p1->next = p2;

x->next = p1;

p1 = t;

}

p1 = p;

struct User\* kill = p1;

while (p1 != NULL)

{

printf("%s %s\n", p1->id, p1->type);

p1 = p1->next;

}

free\_list\_user(kill);

}

if (2 == n) //字典排序输出

{

struct Reader\* head = NULL, \* p, \* p1, \* p2, \* x, \* t;

p = temp\_storage\_reader(head);

p1 = p->next;

p->next = NULL;

while (p1 != NULL)

{

x = p;

p2 = p->next;

while (p2 != NULL && strcmp(p2->id, p1->id) <= 0)

{

x = p2;

p2 = p2->next;

}

t = p1->next;

p1->next = p2;

x->next = p1;

p1 = t;

}

p1 = p;

while (p1 != NULL)

{

printf("%s %s %s %d %d\n", p1->id, p1->company, p1->telephone, p1->can\_borrow, p1->already\_borrow);

p1 = p1->next;

}

struct Reader\* kill = p1;

free\_list\_reader(kill);

}

}

struct Data current\_system\_time(void) //提取系统当前日期

{

struct Data temp;

struct tm\* lt;

time\_t t;

time(&t); //获取Unix时间

lt = localtime(&t); //转为时间结构

temp.year = lt->tm\_year + 1900;

temp.month = lt->tm\_mon + 1;

temp.day = lt->tm\_mday;

return temp;

}

void clear\_buffer(void) //清空缓冲区

{

rewind(stdin);

}

## 测试分析

1. 测试数据

用户文件（部分）

root root Administration

teacher libroot librarian

Amy 123456 reader

John 123456 reader

图书文件（部分）

1 1021 数据库 杨艳 人民邮电 10 4 0 0 0

2 1014 数据结构 赵鹏 高等教育 9 7 0 0 0

3 1106 操作系统 金虎 人民邮电 8 6 2 0 0

1. 调试过程中遇到的问题及解决方法

(1).修改文件内容

解决方案：文件内容全部存入链表，再链表中修改，再文件中覆盖写，达到文件修改效果

(2).用户非法输入

解决方案：判断非法输入并标记检测标记是否有异常，若发现非法输入那系统清空缓冲区并提示用户重新输入。清空缓冲区可以用rewind(stdin);函数

(3).姓氏排序输出（字典排序输出）

解决方案：链表排序可以使用插入排序进行处理

(4).提前当前系统时间

解决方案：使用time.h头文件中的time()函数进行系统时间提取计算

## 设计总结或心得体会

实践是检验真理的唯一标准，程序设计这门课程着重点在于实践，编程能力提高的最好方法就是敲代码，亲手写出完整的程序代码，理论自然重要但是如果没有实践的巩固一切都将是天方夜谭毫无意义。此次程序设计实训项目从项目要求到各个功能设计撰写流程图再到写代码调试全程独立开发完成。对于项目设计概念有了初步的简单了解与实践。从中也收获不少编程技巧与问题处理解决方案