**实验七 综合实验**

**题目：**简易图书管理模拟系统

**班级：**17级计科一班 **姓名：**元雨暄 **学号：**1725111037 **完成日期：**2018.12.28

**一、需求分析**

(1) **输入形式**：输入数字表示执行的操作名，各种操作还需要输入字符串，数字等。

**输入值的范围**：字符、数字

(2)**输出的形式**：输出在界面上以及txt文件中。凹入表打印平衡树在屏幕上。

(3)**功能**：

1、**采编入库：**新购入一种书，经分类和确定书号之后登记到图书帐目中去。如果这种书在帐中已有，则只将该书的总库存量增加。

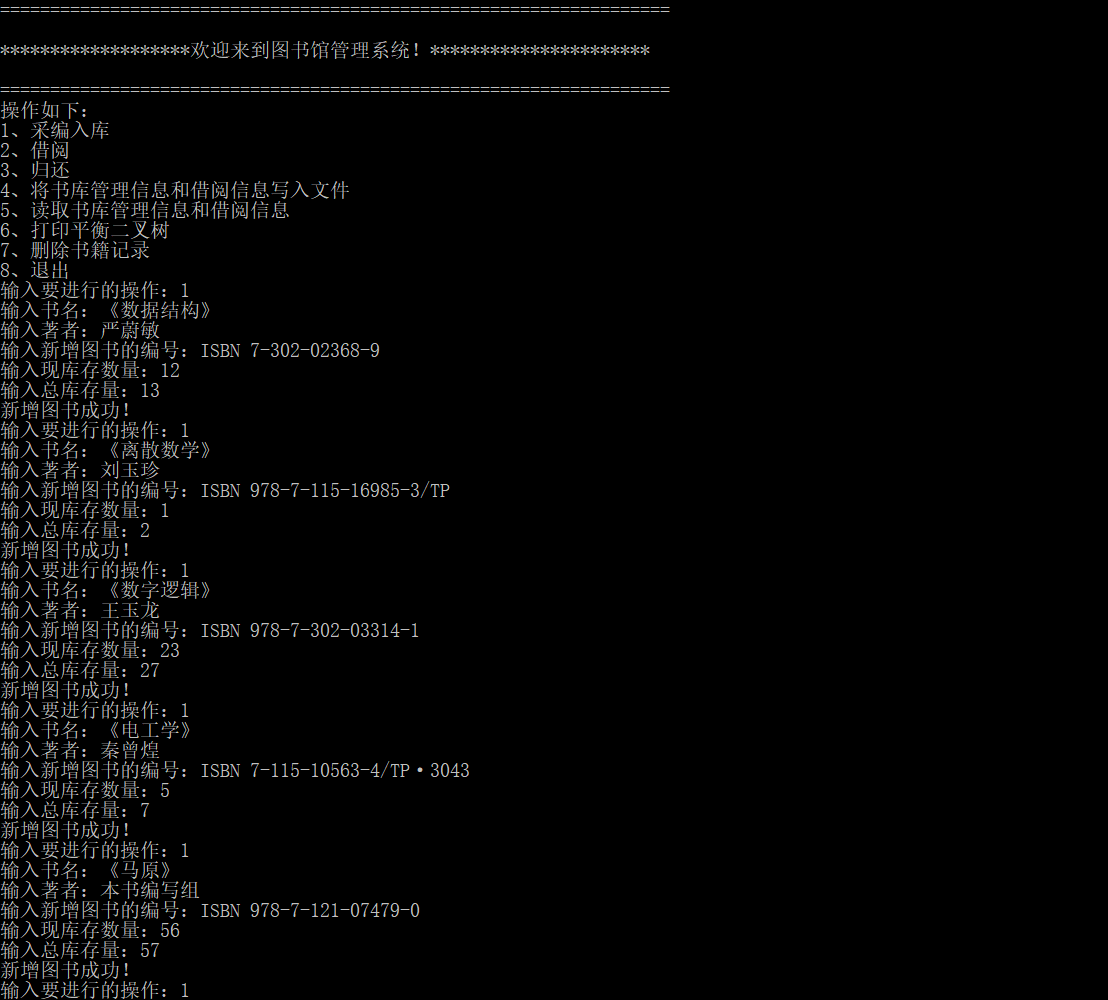
2、**借阅：**如果一种书的现存量大于零，则借出一本，登记借阅者的图书证号和归还期限。

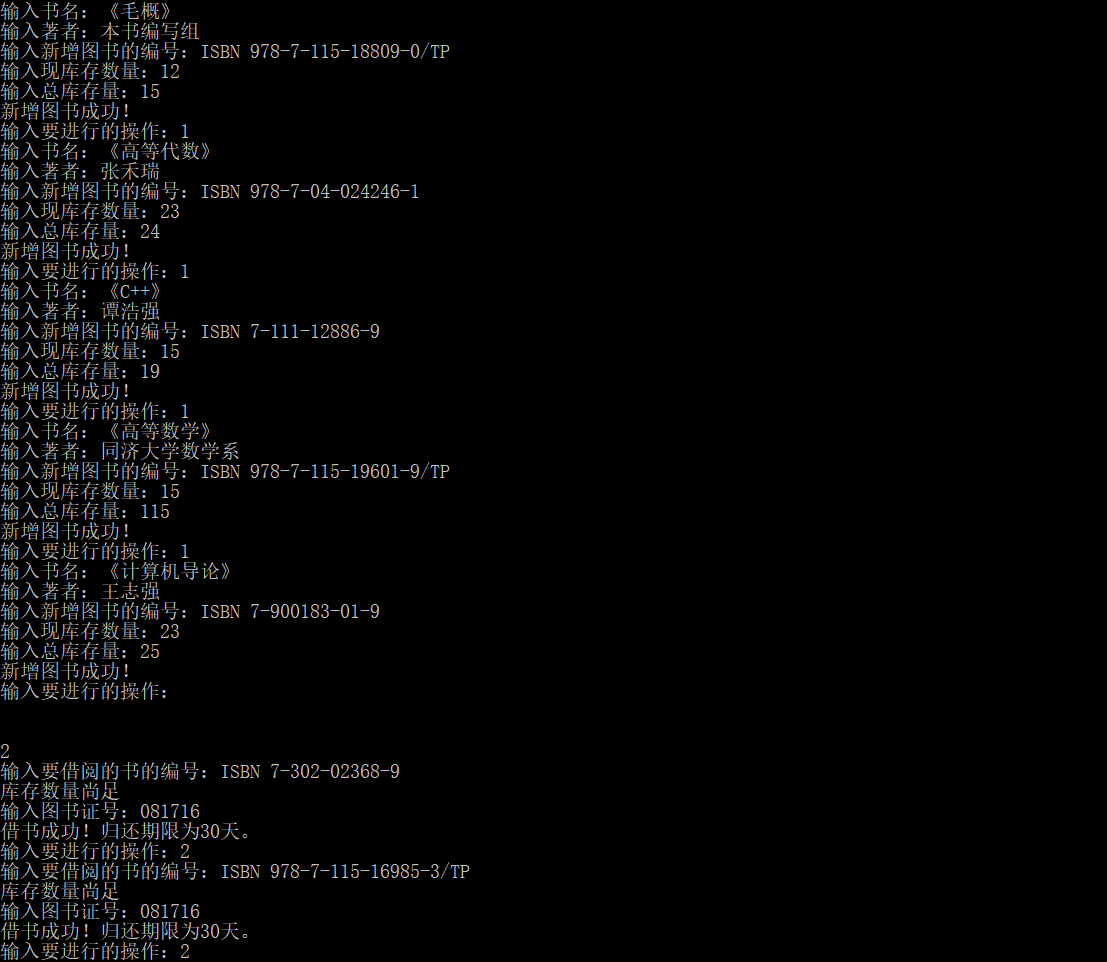
3、**归还：**注销对借阅者的登记，改变该书的现存量（如果借阅者归还所有的书，则注销该借阅者的信息）。

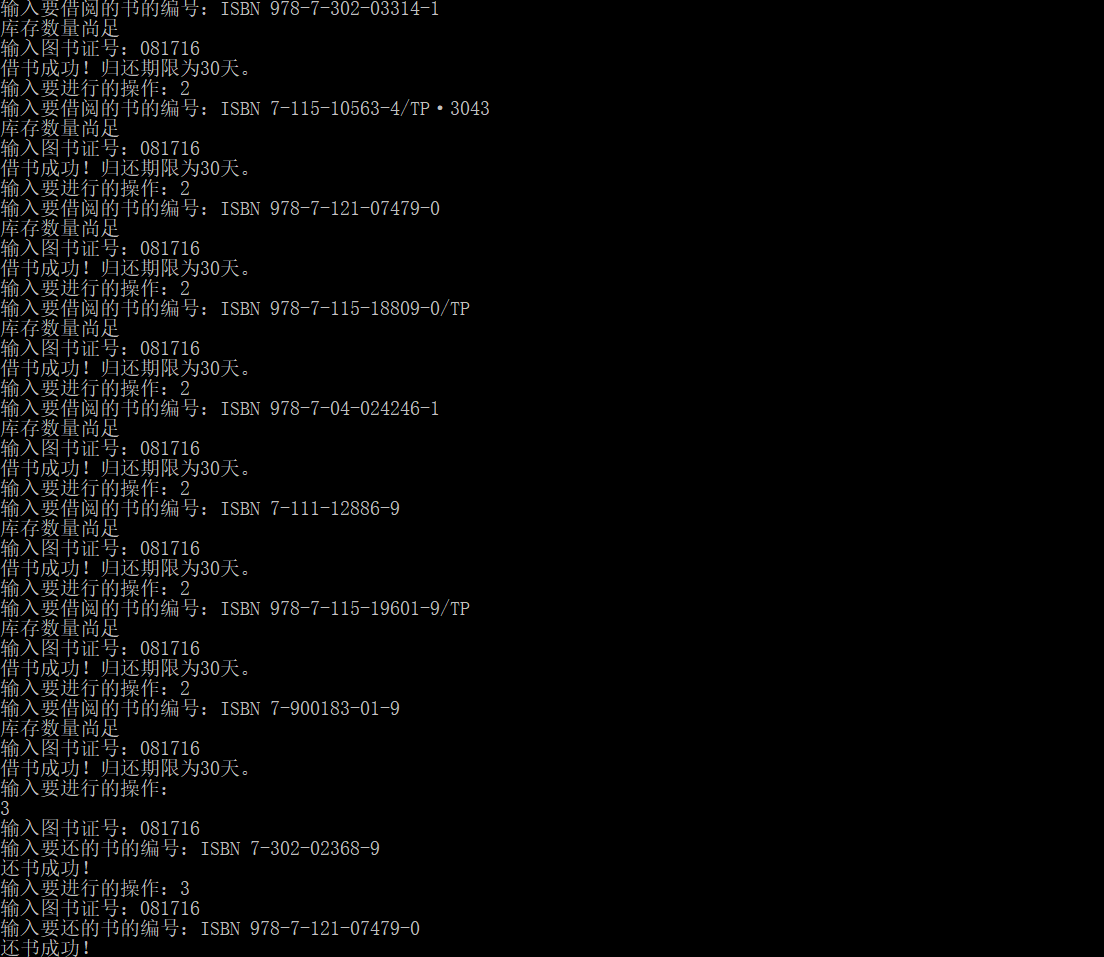
4、**注销：**某种书已无保留价值，将它从图书帐目中注销。

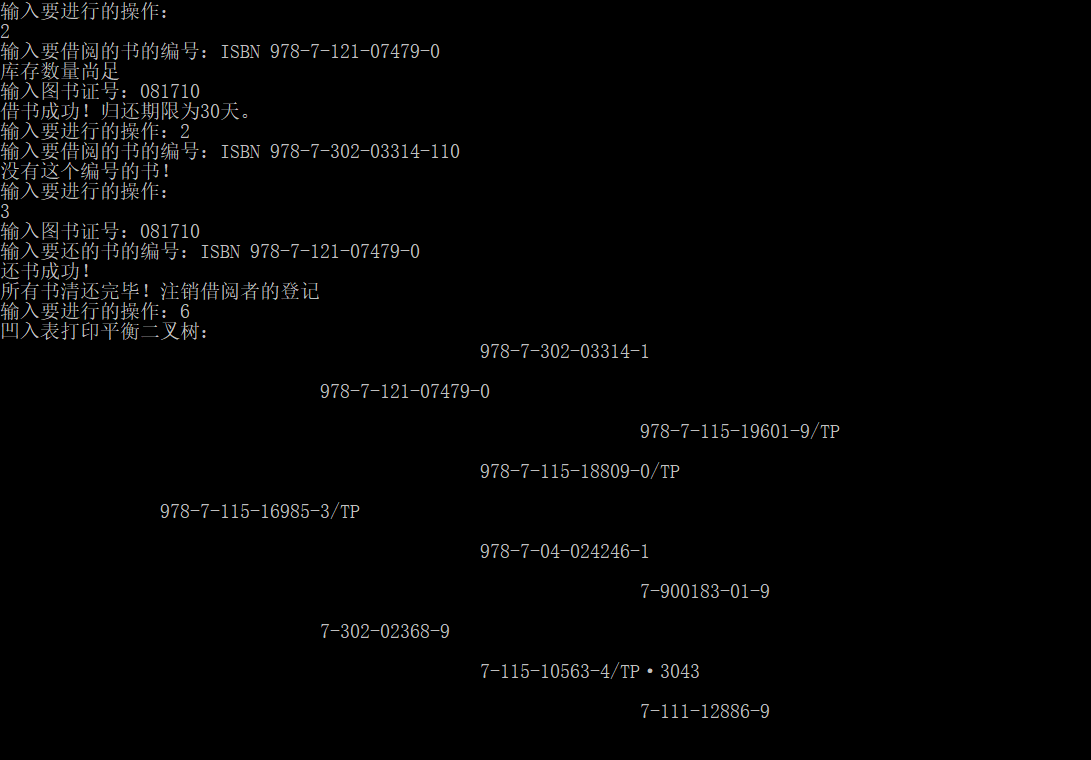
5、**显示：**实现图书管理的显示操作：每次插入或删除平衡二叉树的一个结点之后，更新平衡二叉树的显示。

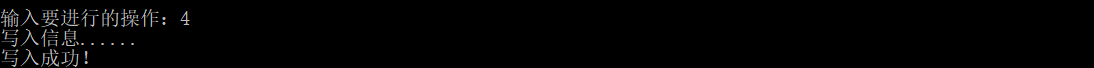
(4)**测试数据：**



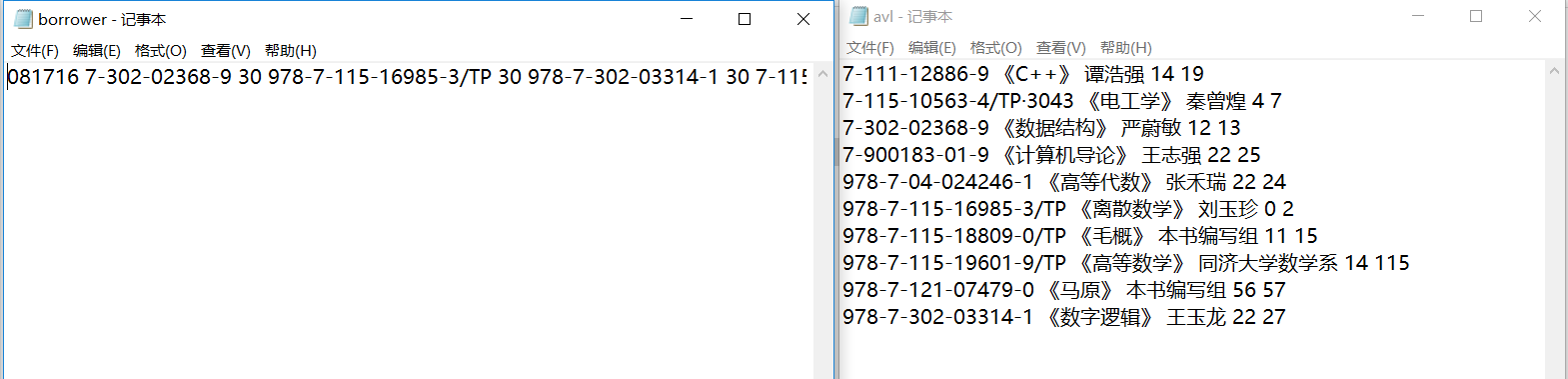








Txt文件展示



**二、概要设计**

(1)抽象数据定义

struct Node //Node 每一本书为一个结点

{

string name, writer; //分别是书名，著者

int quantity; //现有库存

string key; //key为书的编号

int sum; //总库存

int height; //高度

Node \* left;

Node \* right;

};

class AVL //用于记录各种书的数据

{

public:

AVL();

~AVL();

void insert(string key, int n, string na, string p, int all); //用户输入,插入书籍

Node \* find(string key);

void erase(string key);

void print();

void print2txt();

void read\_avl();

void Indentation();

private:

Node \* header;

Node \* ll\_rotate(Node \* y);

Node \* rr\_rotate(Node \* y);

Node \* lr\_rotate(Node \* y);

Node \* rl\_rotate(Node \* y);

void destroy(Node \* node);

int get\_height(Node \* node);

int get\_balance(Node \* node);

Node \* insert\_real(string key, Node \* node, int n, string na, string p, int all);

Node \* find\_real(string key, Node \* node);

Node \* erase\_real(string key, Node \* node); //删除某个结点

void in\_order(Node \* node);

void print\_txt(Node \*node, ofstream &printavl);

void printhelp(Node \*node,string str);

};

class Library

{

public:

Library();

void book\_storing();

void book\_lending();

void book\_returning();

void menu();

void write();

void read();

void print\_avl();

void erase\_book(string s);

private:

struct record

{

string dayleft;

string book\_key;

record \*next;

};

struct borrower

{

string id;

record \*first;

};

AVL avl;

borrower \*br;

};

（2）主程序流程

int main()

{

Library lib;

lib.menu();

system("pause");

return 0;

}

(3)各模块之间的调用

Main（）函数调用menu函数（），通过不同的操作输入，调用Library类对象的各种操作，void book\_storing();void book\_lending();void book\_returning();void menu();void write();void read();void print\_avl();void erase\_book(string s);

主函数模块

Menu函数

其他函数

**三、详细设计**

(1)数据类型的定义

struct Node //Node 每一本书为一个结点

{

string name, writer; //分别是书名，著者

int quantity; //现有库存

string key; //key为书的编号

int sum; //总库存

int height; //高度

Node \* left;

Node \* right;

Node(string key = "") //初始化

{

this->key = key;

this->height = 1;

this->left = this->right = nullptr;

this->name = "";

this->quantity = 0;

this->sum = 0;

this->writer = "";

}

};

class AVL //用于记录各种书的数据

{

public:

AVL();

~AVL();

void insert(string key, int n, string na, string p, int all); //用户输入,插入书籍

Node \* find(string key);

void erase(string key);

void print();

void print2txt();

void read\_avl();

void Indentation();

private:

Node \* header;

Node \* ll\_rotate(Node \* y);

Node \* rr\_rotate(Node \* y);

Node \* lr\_rotate(Node \* y);

Node \* rl\_rotate(Node \* y);

void destroy(Node \* node);

int get\_height(Node \* node);

int get\_balance(Node \* node);

Node \* insert\_real(string key, Node \* node, int n, string na, string p, int all);

Node \* find\_real(string key, Node \* node);

Node \* erase\_real(string key, Node \* node); //删除某个结点

void in\_order(Node \* node);

void print\_txt(Node \*node, ofstream &printavl);

void printhelp(Node \*node,string str);

};

class Library

{

public:

Library();

void book\_storing();

void book\_lending();

void book\_returning();

void menu();

void write();

void read();

void print\_avl();

void erase\_book(string s);

private:

struct record

{

string dayleft;

string book\_key;

record \*next;

};

struct borrower

{

string id;

record \*first;

};

AVL avl;

borrower \*br;

};

(2)操作的伪码算法

Library::Library() //初始化Library

{

br = new borrower[MAX];

for (int i = 0; i < MAX; i++)

{

br[i].id = "";

br[i].first = new record;

}

}

void Library::book\_storing()//存入书籍

{

string s,na,p;

int n,all;

cout << "输入书名：";

cin >> na;

cout << "输入著者：";

cin >> p;

cout << "输入新增图书的编号：ISBN ";

cin >> s;

cout << "输入现库存数量：";

cin >> n;

cout << "输入总库存量：";

cin >> all;

avl.insert(s,n, na, p, all);

cout << "新增图书成功！\n";

//avl.print();

}

void Library::book\_lending() //借出书籍

{

int str2num(string);

string s1;

int j=1;

string id\_num;

cout << "输入要借阅的书的编号：ISBN ";

cin >> s1;

Node \*find = avl.find(s1);

if (!find)

{

cout << "没有这个编号的书！\n";

return;

}

else

{

if (find->quantity <= 0)

{

cout << "库存数量不够了!借书失败。\n";

return;

}

else

{

cout << "库存数量尚足\n输入图书证号：";

cin >> id\_num;

br[0].id = id\_num;

int i = MAX;

while (br[i].id != br[0].id)

i--;

if (i == 0) //说明之前没有借阅记录

{

for (j = 1; j < MAX; j++)//!!!

{

if (br[j].id=="")

{

br[j].id = id\_num;

br[j].first->book\_key = s1;

br[j].first->dayleft = "30";

br[j].first->next = NULL;

avl.find(s1)->quantity--;

break;

}

}

}

else //有借阅记录 (还要看有无超期未归还的书

{

record \*r =br[j].first;

while (r->next)

{

r = r->next;

if (str2num(r->dayleft) <= 0)

{

cout << "有超期未归还的书，借书失败，请先还书！\n";

return;

}

}

avl.find(s1)->quantity--;

record \*e = new record;

e->book\_key = s1;

e->dayleft = "30";

e->next = NULL;

r->next = e;

}

cout << "借书成功！归还期限为30天。\n";

return;

}

}

}

void Library::book\_returning()//归还书籍

{

string m;

int i;

string s2;

record \*c;

cout << "输入图书证号：";

cin >> m;

cout << "输入要还的书的编号：ISBN ";

cin >> s2;

for (i = 1; i <MAX; i++)

{

if (br[i].id == m)

{

c = br[i].first;

if (c->book\_key == s2)

{

avl.find(s2)->quantity++;

if (c->next)

{

c = c->next;

cout << "还书成功！\n";

break;

}

else

{

cout << "还书成功！\n所有书清还完毕！注销借阅者的登记\n";

br[i].id = "";

br[i].first = NULL; //c=NULL;

break;

}

}

while (c)

{ //find 并且加上quantity

if (c->next->book\_key == s2)

{

record \*link = c->next;

c->next = link->next;

avl.find(s2)->quantity++;

cout << "还书成功！\n";

return;

}

else c = c->next;

}

}

}

}

void Library::write()

{

avl.print2txt();

ofstream outbr("borrower.txt");

for (int i = 1; i < MAX; i++)

{

if (br[i].id != "")

{

record \*f;

outbr <<br[i].id << " "; //借书证号

f = br[i].first;

while (f)

{

outbr <<f->book\_key <<" "<< f->dayleft <<" "; //书号，归还期限

f = f->next;

}

outbr << "#"<<endl;

}

}

outbr.close();

}

void Library::read() //读入

{

avl.read\_avl();

ifstream inbr;

inbr.open("borrower.txt");

int k = 1,step = 0;

string g;

while (!inbr.eof())

{

inbr >> g;

br[k].id = g;

record \*rd = br[k].first;

while (g != "#")

{

inbr >> g;

if (rd->dayleft=="")

{

if (step == 0)

{

br[k].first->book\_key = g;

step++;

}

else

{

br[k].first->dayleft = g;

step = 0;

}

}

else

{

record \*nex = new record;

if (nex->dayleft == "")

{

if (step == 0)

{

nex->book\_key = g;

step++;

}

else

{

nex->dayleft = g;

step = 0;

rd->next = nex;

rd = rd->next;

nex->next = NULL;

}

}

}

}

k++;

}

inbr.close();

}

void Library::print\_avl()

{

avl.Indentation();

}

void Library::erase\_book(string s)

{

avl.erase(s);

}

void Library::menu()

{

char ch;

string f;

cout<<"==================================================================="<<endl;

cout<<"\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*欢迎来到图书馆管理系统！\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"<<endl;

cout<<"==================================================================="<<endl;

cout<<"操作如下：\n1、采编入库\n2、借阅\n3、归还\n4、将书库管理信息和借阅信息写入文件\n5、读取书库管理信息和借阅信息\n6、打印平衡二叉树\n7、删除书籍记录\n8、退出\n";

ch = '0';

while (ch!='4')

{

cout << "输入要进行的操作：";

cin >> ch;

switch (ch)

{

case '1':

book\_storing();

continue;

case '2':

book\_lending();

continue;

case '3':

book\_returning();

continue;

case '4':

cout << "写入信息......\n";

write();

cout << "写入成功！\n";

continue;

case '5':

cout << "读取信息......\n";

read();

cout << "读取成功！\n";

continue;

case '6':

cout << "凹入表打印平衡二叉树：\n";

print\_avl();

cout << endl;

continue;

case '7':

cout << "输入需要删除记录的书的书号：ISBN ";

cin >> f;

erase\_book(f);

cout << "删除成功！\n";

continue;

case '8':

cout << "退出成功！欢迎下次再来！" << endl;

return;

}

}

}

(3)函数的调用关系图

=

Menu函数

Book\_storing()book\_lending()book\_returning()write() read()print\_avl() erase\_book(f)

Avl.insert() avl.find() avl.find() avl.print2txt() avl.read\_avl() avl.Indentation()avl.erase()

Insert\_real() print\_txt() printhelp() erase\_real()

**四、调试分析**

**(1)问题：**将文件读取并且再次用已有的图书证号借阅图书后无法再次写入文件，写入文件出错。尚未解决。

**(2)算法的时空分析：**

**(1)时间复杂度：设n为结点个数**

**AVL类**

Node \* ll\_rotate(Node \* y) **O(1)**

Node \* rr\_rotate(Node \* y) **O(1)**

Node \* lr\_rotate(Node \* y) **O(1)**

Node \* rl\_rotate(Node \* y) **O(1)**

void destroy(Node \* node) **O(n)**

int get\_height(Node \* node)**O(1)**

int get\_balance(Node \* node)**O(1)**

Node \* insert\_real(string key, Node \* node, int n, string na, string p, int all) **O(logn)**

Node \* find\_real(string key, Node \* node)**O(logn)**

Node \* erase\_real(string key, Node \* node); //删除某个结点**O(logn)**

void in\_order(Node \* node) **O(n)**

void print\_txt(Node \*node, ofstream &printavl) **O(n)**

void printhelp(Node \*node,string str) **O(n)**

**Library类**

void book\_storing()**O(logn)**

void book\_lending()**O(logn)**

void book\_returning()**O(logn)**

void menu()**O(n)**

void write()**O(1)**

void read()**O(1)**

void print\_avl()**O(1)**

void erase\_book(string s) **O(1)**

**(2)空间复杂度：**

**AVL类**

Node \* ll\_rotate(Node \* y) **O(1)**

Node \* rr\_rotate(Node \* y) **O(1)**

Node \* lr\_rotate(Node \* y) **O(1)**

Node \* rl\_rotate(Node \* y) **O(1)**

void destroy(Node \* node) **O(1)**

int get\_height(Node \* node)**O(1)**

int get\_balance(Node \* node)**O(1)**

Node \* insert\_real(string key, Node \* node, int n, string na, string p, int all) **O(1)**

Node \* find\_real(string key, Node \* node)**O(1)**

Node \* erase\_real(string key, Node \* node); //删除某个结点**O(1)**

void in\_order(Node \* node) **O(1)**

void print\_txt(Node \*node, ofstream &printavl) **O(1)**

void printhelp(Node \*node,string str) **O(1)**

void in\_order(Node \* node) **O(n)**

void print\_txt(Node \*node, ofstream &printavl) **O(n)**

void printhelp(Node \*node,string str) **O(n)**

**Library类**

void book\_storing()**O(1)**

void book\_lending()**O(1)**

void book\_returning()**O(1)**

void menu()**O(1)**

void write()**O(1)**

void read()**O(1)**

void print\_avl()**O(1)**

void erase\_book(string s) **O(1)**

**五、用户使用说明**

当弹出界面时，用户可以根据菜单的提示，输入相应操作的数字，根据每次操作弹出的提示完成操作。用户需要输入字符串等。

**六、测试结果**

第一部分已经展示了用户自己输入相应信息并将图书记录和借阅记录分别写入相应的txt文件

这里展示一下读入txt文件。

