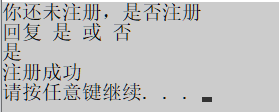
## 需求分析与社会价值

这代码实现了一个简单的图书管理系统，具备基本的图书录入、显示、删除、查找、借阅、归还和保存功能。社会价值在于方便图书管理，提高图书馆工作效率。

## 实现功能（列表分别阐述实现什么具体功能，以及如何正确运行相应功能，需要附上运行截图）

**简单注册功能（0）**



**增加图书（1）**

输入书名、数量、编号，将图书信息添加到系统中。

文本

描述已自动生成

**显示图书信息（2）**

列出系统中所有图书的书名、数量和编号。

图示, 示意图

描述已自动生成

**删除图书（3）**

根据输入的编号删除相应的图书。

图示

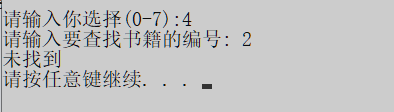
低可信度描述已自动生成

**查找图书（4）**

根据输入的编号查找图书信息。

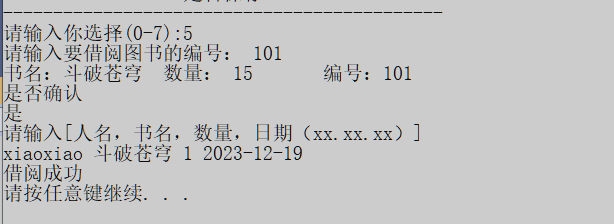
图示, 文本

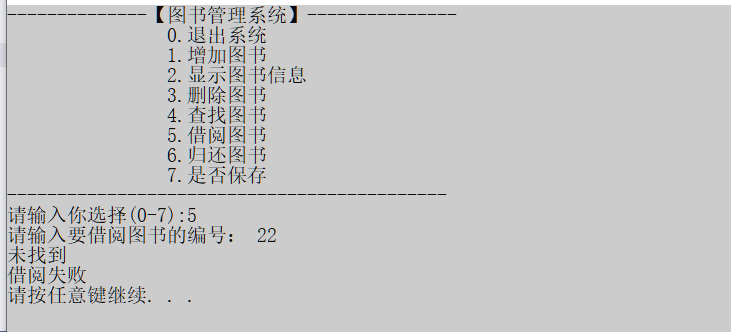
描述已自动生成



**借阅图书（5）**

根据输入的编号查找图书信息，确认后输入人名、书名、数量、日期，将借阅信息添加到系统中。





**归还图书（6）**

根据输入的人名（实际是书的编号）查找借阅信息，确认后将该借阅信息从系统中删除。

图形用户界面, 文本, 应用程序

描述已自动生成

**是否保存（7）**

将系统中的图书和借阅信息保存到文件中。

文本

描述已自动生成

## C++知识点（列表分别结合局部代码举例描述，需要描述该知识点内容和解决的功能需求）

**1.类与对象**

使用class定义Book、Person和Library类，实现面向对象的图书管理系统。

**2.容器类 vector**

使用vector存储多个Book和 Person对象。

**3.文件操作 ofstream**

使用ofstream将图书和借阅信息保存到文件中。

**4.Lambda 表达式**

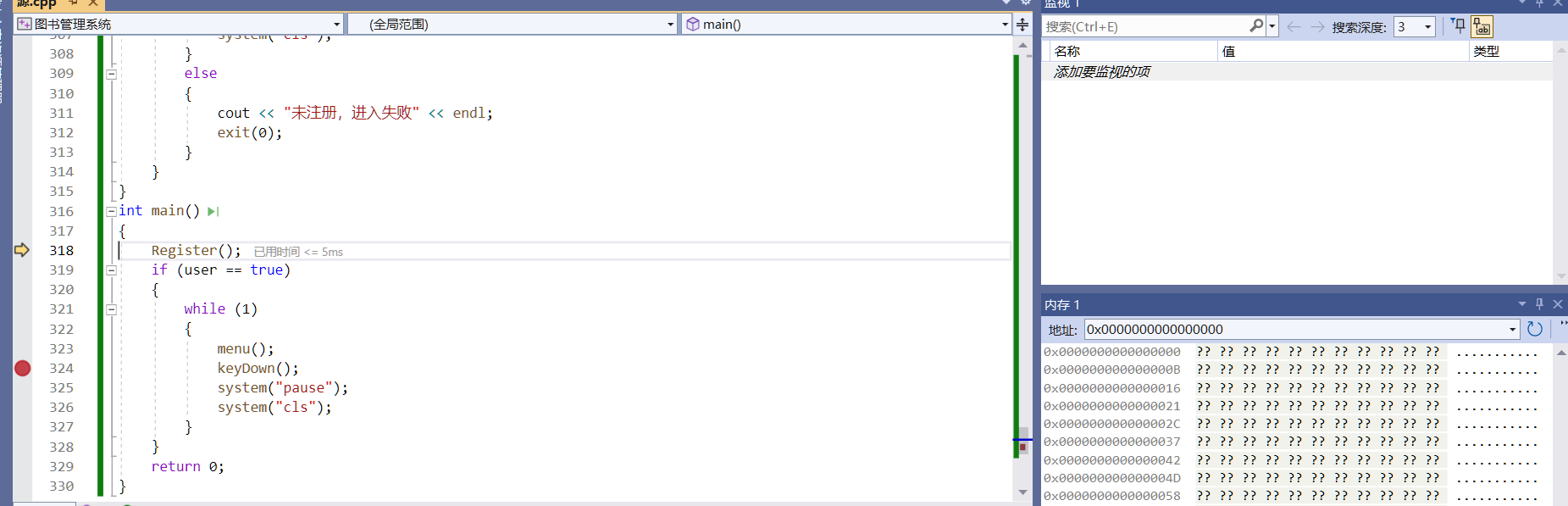
在删除图书和归还图书的功能中使用 Lambda 表达式。

**5.STL 算法**

使用 STL 算法中的 remove\_if 删除指定元素。

## 代码调试案例 （结合截图展示如何用代码调试工具一步步查找错误和改正错误）

在vs2022中按下F10进行调试



图形用户界面, 应用程序, Word

描述已自动生成

## 源代码 （需要附上代码行数）

**330行**

#define \_CRT\_SECURE\_NO\_WARNINGS 1

#include<iostream>

#include <cstring>

#include <algorithm>

#include<string>

#include<vector>

#include<fstream>

using namespace std;

bool user = false;//注册信息

class Book

{

public:

Book(string name, int count, string Isbn)

{

this->name = name;

this->count = count;

this->IBSN = Isbn;

}

void setName(string name)

{

this->name = name;

}

void setCount(int count)

{

this->count = count;

}

string& getName()

{

return name;

}

int& getCount()

{

return count;

}

void setIbsn(string& ibsn)

{

this->IBSN = ibsn;

}

string& getIbsn()

{

return IBSN;

}

private:

string name;//书名

int count;//数量

string IBSN;//编号

};

class Person

{

public:

Person(string Pname, int count, string Bname, string data)

{

this->Bname = Bname;

this->count = count;

this->Pname = Pname;

this->data = data;

}

string& getBname()

{

return Bname;

}

string& getPname()

{

return Pname;

}

string& getData()

{

return data;

}

int& getCount()

{

return count;

}

private:

string Pname;//人名

int count;//数量

string Bname;//书名

string data;

};

class Library

{

public:

void addBook(const Book& book)

{

B.push\_back(book);

}

void removeBookByIbsn(const string& ibsn)

{

if (!B.empty())

{

for (auto& v : B)

{

if (v.getIbsn() == ibsn)

{

B.erase(remove\_if(B.begin(), B.end(),

[&](Book& book) { return book.getIbsn() == ibsn; }),

B.end());

}

}

}

else

{

cout << "库存为0" << endl;

}

}

void listBooks()

{

if (B.empty())

{

cout << "库存为0" << endl;

}

for (auto& b : B)

{

cout << "书名：" << b.getName() << "\t数量： " << b.getCount() << "\t编号：" << b.getIbsn() << endl;

}

}

bool seacrchBook(string ISBN)

{

for (auto& b : B)

{

if (b.getIbsn() == ISBN)

{

cout << "书名：" << b.getName() << "\t数量： " << b.getCount() << "\t编号：" << b.getIbsn() << endl;

return true;;

}

}

cout << "未找到" << endl;

return false;

}

void borrow(string ISBN)

{

if (seacrchBook(ISBN))

{

cout << "是否确认" << endl;

string key;

cin >> key;

if (key == "是")

{

string m\_Pname;

string m\_Bname;

string m\_data;

int m\_count;

cout << "请输入[人名，书名，数量，日期（xx.xx.xx）]" << endl;

cin >> m\_Pname >> m\_Bname >> m\_count >> m\_data;

P.push\_back({ m\_Pname,m\_count,m\_Bname,m\_data });

cout << "借阅成功" << endl;

}

}

else

{

cout <<"借阅失败" << endl;

}

}

void Return(string ISBN)

{

if (!P.empty())

{

for (auto& v : P)

{

if (v.getPname() == ISBN)

{

P.erase(remove\_if(P.begin(), P.end(),

[&](Person& p) { return p.getPname() == ISBN; }),

P.end());

cout<<"归还成功" << endl;

}

else

{

cout << "未找到" << endl;

}

}

}

else

{

cout << "没有要归还的书籍" << endl;

}

}

void saveToFile(const string& filename)

{

ofstream outputFile(filename);

// Save book information

for (auto& book : B)

{

outputFile << "Book|" << book.getName() << "|" << book.getCount() << "|" << book.getIbsn() << endl;

}

// Save person information

for (auto& person : P)

{

outputFile << "Person|" << person.getPname() << "|" << person.getCount() << "|" << person.getBname() << "|" << person.getData() << endl;

}

outputFile.close();

cout << "Data saved to file: " << filename << endl;

}

private:

vector<Book> B;

vector<Person> P;

};

Library library;//全局变量

void menu()

{

printf("--------------【图书管理系统】---------------\n");

printf("\t\t0.退出系统\n");

printf("\t\t1.增加图书\n");

printf("\t\t2.显示图书信息\n");

printf("\t\t3.删除图书\n");

printf("\t\t4.查找图书\n");

printf("\t\t5.借阅图书\n");

printf("\t\t6.归还图书\n");

printf("\t\t7.是否保存\n");

printf("--------------------------------------------\n");

printf("请输入你选择(0-7):");

}

void addBook()

{

string ch;

int num;

string Isbn;

cin >> ch >> num >> Isbn;

library.addBook({ ch,num,Isbn });

cout << "录入成功" << endl;

}

void showBook()

{

library.listBooks();

}

void RomveBook()

{

string nAme;

cin >> nAme;

library.removeBookByIbsn(nAme);

cout << "删除成功" << endl;

}

void SeacrchBook()

{

string ISBN;

cin >> ISBN;

library.seacrchBook(ISBN);

}

void Borrow()

{

string ISBN;

cin >> ISBN;

library.borrow(ISBN);

}

void returnBook()

{

string ISBN;

cin >> ISBN;

library.Return(ISBN);

}

void keyDown()

{

int select = 0;

cin >> select;

switch (select)

{

case 0:

exit(0);

break;

case 1:

cout << "请输入图书信息（书名，数量,编号）" << endl;

addBook();

break;

case 2:

showBook();

break;

case 3:

cout << "请输入要删除书籍的编号: ";

RomveBook();

break;

case 4:

cout << "请输入要查找书籍的编号: ";

SeacrchBook();

break;

case 5:

cout << "请输入要借阅图书的编号： ";

Borrow();

break;

case 6:

cout << "请输入要归还人的姓名： ";

returnBook();

break;

case 7:

library.saveToFile("save.txt");

break;

}

}

//阿甘正传 10 101

void Register()

{

if (user == false)

{

cout << "你还未注册，是否注册" << endl;

cout << "回复 是 或 否 " << endl;

string select;

cin >> select;

if (select == "是")

{

cout << "注册成功" << endl;

user = true;

system("pause");

system("cls");

}

else

{

cout << "未注册，进入失败" << endl;

exit(0);

}

}

}

int main()

{

Register();

if (user == true)

{

while (1)

{

menu();

keyDown();

system("pause");

system("cls");

}

}

return 0;

## }思考及学习心得

这部分可以包括对整个项目的思考，遇到的困难及解决方法，学到的新知识，以及对代码的改进建议等。

总体来说，这个项目是一个简单但完整的图书管理系统。在实现功能的基础上，可以考虑进一步优化代码结构、增加异常处理机制等，使系统更加健壮和用户友好。